Software Requirements Specification

UAV Swarm

CS 490, SE 450, CEC 420, FALL 2020

Team Name: Swarmiez

Team Members:

* Samantha Balistreri
* Tyler Wise
* Joseph Moran
* Michael Fornito
* Will Edwards
* Daniela Regueira

Embry-Riddle Aeronautical University

Version 2

# Table of Contents

[**Table of Contents**](#_m1p5cknrnfvd) **2**

[**Section 1: Introduction**](#_z61y4x1qdolt) **3**

[1.1 System to be Produced:](#_h31ybe6jp5pn) 3

[1.2 Applicable Standards](#_xrtgtui3tm6x) 3

[1.3 Definitions, Acronyms, and Abbreviations](#_pzodhv4kc2ex) 3

[**Section 2: Product Overview**](#_xu1rshw3xzjm) **3**

[2.1 Assumptions:](#_v2vvcdeed4xe) 3

[2.2 Stakeholders:](#_nhwsdzsm97ar) 4

[2.3 Event Table:](#_tlvvvvsq9lug) 4

[2.4 Use Case Diagram](#_857mr7saez5j) 5

[2.5 Use Case Descriptions:](#_l0kuzpeojvq1) 6

[**Section 3: Specific Requirements**](#_x18wax9tluy5) **6**

[3.1 Functional Requirements](#_hiwtdbcsidte) 6

[3.2 Interface Requirements](#_ojtbmqdezaba) 8

[3.3 User and Human Factors Requirements](#_24xogj971hg5) 10

[3.4 Documentation Requirements](#_csea7zcp85ly) 19

[3.5 Data Requirements](#_jtffgwn70b74) 20

[3.6 Resource Requirements](#_j7ewxmrm8p8o) 22

[3.7 Security Requirements](#_dk6z05iosq00) 22

[3.8 Quality Assurance Requirements](#_1kdk3s9yl3mk) 22

[**Section 4: Supporting Material**](#_upvy14ljl8si) **23**

# [Section 1: Introduction](#_15bjjs2adel8)

## 1.1 System to be Produced:

* The product to be produced will be a Swarm Behavior Network that will allow a swarm of no less than 10 drones fly in swarming formations.The drones will use image processing and swarming algorithms to accomplish this goal.This product will be done strictly in a simulated environment and there is no physical components to this product.Reference:https://sites.google.com/view/kavehfathian/code/vision-based-formation-control-in-unreal-engine-4-airsim

## 1.2 Applicable Standards

* This project should be held to industry standards in both the engineering industry as well as the aviation industry

## 1.3 Definitions, Acronyms, and Abbreviations

* UAV- Unmanned Aerial Vehicle
* UAS - Unmanned Aircraft Systems
* UE4 - Unreal Engine 4
* MVS - Microsoft Visual Studio
* SBF - Swarm Behavior Framework
* AirSim - An open-source, cross platform simulator for drones, ground vehicles such as cars and various other objects, built on Epic Games’ Unreal Engine 4 as a platform for AI research.

# Section 2: Product Overview

## 2.1 Assumptions:

* Assume that the user will have no control over the drones and it will be strictly autonomous
* Assume that the swarm framework will only work in the UE4/AirSim environment
* Assume that the drone swarm is no less than ten drones

## 2.2 Stakeholders:

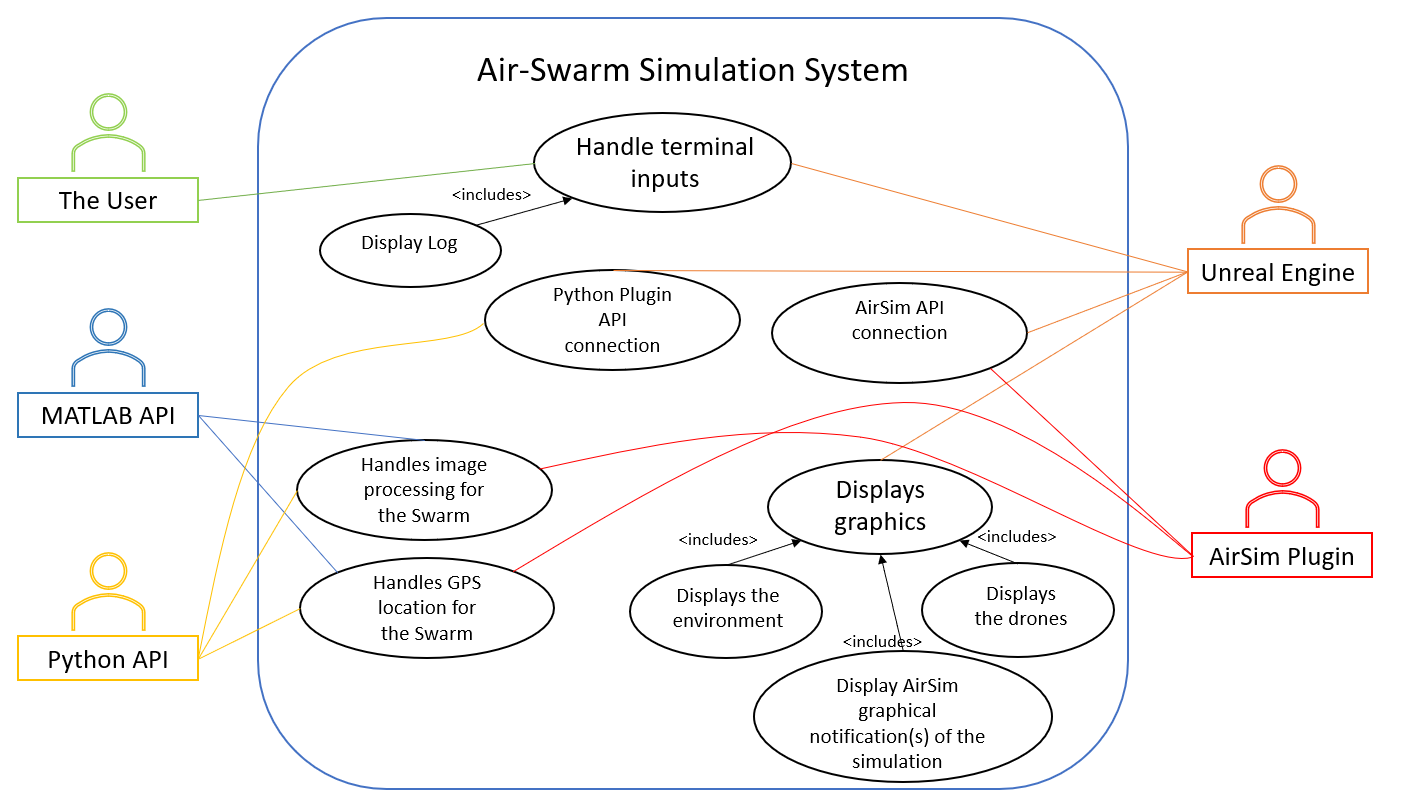
* Customer - Dr. Akbas
  + Identifies requirements and is invested in the final product
* Product Owner - Juan
  + Assisting the scrum team in prioritization and evaluating project progress
* Scrum Team Members - Samantha, Tyler, Joseph, Michael, Will, and Daniela
  + Completion of the project

## 2.3 Event Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Event Name | External Stimuli | External Responses | Internal data and state |
| Start | User clicks button to start the simulation | Simulation displays the running simulation | Algorithm begins running |
| Pause | User clicks the button to pause the simulation | Simulation stops moving but simulation is not reset | Algorithm pauses and can continue to run when prompted |
| Stop | User clicks button to stop the simulation | Simulation is no longer running and UI resets | Algorithm stops running and resets |
| Image | User calls for the python function for the image-based simulation mode | Terminal displays the swarm utilizing the image-based algorithm | Runs algorithm that is appropriate for image-based swarming |
| Location | User calls for the python function for the location-based simulation mode | Terminal displays the swarm utilizing the image-based algorithm | Runs algorithm that is appropriate for location-based swarming |
| Add Drone | User calls for the python function to add the drone in the swarm | Terminal displays if it was successful | Modifies settings file |
| Remove Drone | User calls for the python function to remove the drone from the swarm | Terminal displays if it was successful | Modifies settings file |

## 

## 2.4 Use Case Diagram



## 2.5 Use Case Descriptions:

1. Display graphics – This System (Unreal Engine) shall display the necessary graphical information to the user
2. Display the drones – The System shall display the drones via Unreal Engine and the player object from AirSim
3. Display the Environment – The System shall (Unreal Engine) shall display the environment from 3rd party marketplace
4. Display the AirSim notifications – The System shall display graphical notifications automatically provided by the AirSim API
5. Handles Image Processing of the Swarm – MATLAB and Python shall be used to handle the computation of the swarm so that it may be processed for AirSim to the Unreal Engine
6. Handles GPS Location of the Swarm – MATLAB and Python shall be used to handle the computation of the swarm so that it may be processed for AirSim to the Unreal Engine
7. Handle Terminal Inputs – The user shall call their desired python functions
8. Display Log**-** The terminal will output to the log file
9. Python Plugin API Connection – Unreal shall connect the python script pythons so that it may be accessed from the terminal
10. AirSim API Connection – AirSim shall connect to Unreal so that the AirSim can use the Unreal Functions

# Section 3: Specific Requirements

## 3.1 Functional Requirements

|  |
| --- |
| No: F1 |
| Statement: The program shall let the user select the amount of drones in the simulation. |
| Source: Basic functions |
| Dependency: None |
| Conflicts: The user cannot select a number greater than 10 or a number that is 0 or lower. The number will be an integer. |
| Evaluation Method: The number of selected drones from the user in the interface will be shown in the simulation. |
| Revision History: Daniela Regueira - 09/25/2020 - Version 1 |

|  |
| --- |
| No: F2 |
| Statement: The program shall allow the user to start the simulation. |
| Source: Basic functions |
| Dependency: F1 |
| Conflicts: None |
| Evaluation Method: The simulation with the selected number of drones will start. |
| Revision History: Daniela Regueira - 09/25/2020 - Version 1 |

## 

|  |
| --- |
| No: F3 |
| Statement: The program shall allow the user to stop the simulation. |
| Source: Basic functions |
| Dependency: F2 |
| Conflicts: None |
| Evaluation Method: The simulation will end. |
| Revision History: Daniela Regueira - 09/25/2020 - Version 1 |

## 

## 

|  |
| --- |
| No: F4 |
| Statement: The program should allow the user to control the swarming algorithm |
| Source: Extra functions |
| Dependency: F2, F1 |
| Conflicts: Only one formation is allowed to be selected. Default formation is “Air Space Optimization”. |
| Evaluation Method: The simulation will run the formation selected. |
| Revision History: Daniela Regueira - 09/25/2020 - Version 1  Tyler Wise - 10/28/2020 - Updated to allow control of swarming algorithm to include extensibility. |

## 

## 

## 3.2 Interface Requirements

|  |
| --- |
| No: #I1 |
| Statement: The camera on each drone shall broadcast its current image to the sink drones no less than every 5 seconds. This data shall be encoded as a byte array. |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The message with the raw data is received within a 5 second interval. |
| Revision History:   1. Created by Tyler Wise, 9/24/20 |

|  |
| --- |
| No: #I2 |
| Statement: Each sink drone shall distribute the relative location of each drone to itself to all other drones within a 5 second interval. This data shall be encoded as a dictionary of guids to vectors. |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The message with the data is received within a 5 second interval. 2. The number of elements in the list is equal to the number of drones in the simulation. |
| Revision History:   1. Created by Tyler Wise, 9/24/20 2. Updated data types, TJW, 10/29/20 |

## 3.3 User and Human Factors Requirements

* The following are user interface requirements and system interface requirements that will allow the user to control how they want to operate the UAV drones in the simulation.

|  |
| --- |
| No: #HF1 |
| Statement: User shall call the python function for the simulation mode such as either Image Processing and GPS navigation mode |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. User calls the python function in the terminal for Image Processing or GPS navigation 2. Notify the user within the terminal what mode will be executed |
| Revision History:   1. Created by Michael Fornito, 9/22/200 2. Modified on 9/24/2020, Evaluation Method 3. Modified on 10/29/20, Eval Method & Statement |

|  |
| --- |
| No: #HF2 |
| Statement: User shall be able to start the simulation |
| Source: <source of the requirement> |
| Dependency:   1. The mode must exist within the settings file for startup, HF1 2. The number of drones must exist in the settings file |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The dependencies stated in HF2 have been inputted    1. If not the simulation will not start 2. User clicks start 3. The desired mode selected will be executed and notify the user 4. The system will display a visual representation of the swarm |
| Revision History:   1. Created by Michael Fornito, 9/22/200 2. Modified on 9/24/2020, Dependency 3. Modified on 10/29/2020, Evaluation Method and Dependency |

|  |
| --- |
| No: #HF3 |
| Statement: User shall be able to pause the simulation |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method: The drones stop their simulation based on the desired selected mode |
| Revision History:   1. Created by Michael Fornito, 9/22/20 2. Modified on 9/24/2020, Evaluation Method |

|  |
| --- |
| No: #HF4 |
| Statement: User shall be able to end the simulation |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The drones stop swarming and the execution of the program has ended |
| Revision History:   1. Created by Michael Fornito, 9/22/20 2. Modified on 9/24/2020, Evaluation Method |

|  |
| --- |
| No: #HF5 |
| Statement: User shall be notified which drone(s) have failed during the simulation |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user has started the simulation, ensure #HF2 is followed 2. One or more of the drones crash 3. The user is notified during the simulation that the drone(s) have crashed through the terminal |
| Revision History:   1. Created by Michael Fornito, 9/22/20 2. Modified on 9/24/2020, Evaluation Method 3. Modified on 10/29/2020, Evaluation Method |

|  |
| --- |
| No: #HF6 |
| Statement: User shall be notified which drone(s) have successfully finished the simulation |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user has started the simulation, ensure #HF2 is followed 2. The drones do not crash during the entire simulation run 3. The user is notified at the end of the simulation that it was successful through the terminal |
| Revision History:   1. Created by Michael Fornito, 9/22/20 2. Modified on 9/24/2020, Evaluation Method 3. Modified on 10/29/2020, Evaluation Method |

|  |
| --- |
| No: #HF7 |
| Statement: User shall be able to see the perspective of the selected drone flying one at a time |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: At least one or more drones must be inputted |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user has started the simulation, ensure #HF2 is followed 2. The user can select the drone they want to see 3. The selected drone’s POV is displayed to the user |
| Revision History:   1. Created by Michael Fornito, 9/22/20 2. Modified on 9/24/2020, Evaluation Method 3. Modified on 9/25/2020, Conflicts |

|  |
| --- |
| No: #HF8 |
| Statement: User shall input the distribution shape of the drones |
| Source: <source of the requirement> |
| Dependency:   1. TBD - depends on the algorithm (more research needed) |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user shall call the python function to change the distribution size 2. The user is notified that it was successfully changed or not 3. The user clicks the start button 4. The simulation shall execute with the given size |
| Revision History:   1. Created by Michael Fornito, 9/28/20 2. Modified on 10/29/2020, Evaluation Method |

|  |
| --- |
| No: #HF8 |
| Statement: User shall input the distribution shape of the drones |
| Source: <source of the requirement> |
| Dependency:   1. TBD - depends on the algorithm (more research needed) |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user shall call the python function to change the distribution size 2. The user is notified that it was successfully changed or not 3. The user clicks the start button 4. The simulation shall execute with the given size |
| Revision History:   1. Created by Michael Fornito, 9/28/20 2. Modified on 10/29/2020, Evaluation Method |

|  |
| --- |
| No: #HF9 |
| Statement: User shall call the python function to add a drone |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user imports the modify settings package 2. The user calls the add drone function 3. The terminal shall tell the user if it was successful or not 4. The settings were changed based on the input 5. The user has started the simulation, ensure #HF2 is followed 6. The simulation shall display the correct inputted # of drones |
| Revision History:  Created by Michael Fornito, 10/29/2020 |

|  |
| --- |
| No: #HF10 |
| Statement: User shall call the python function to remove a drone |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user imports the modify settings package 2. The user calls the remove drone function 3. The terminal shall tell the user if it was successful or not 4. The settings were changed based on the input 5. The user has started the simulation, ensure #HF2 is followed 6. The simulation shall display the correct # of drones |
| Revision History:  Created by Michael Fornito, 10/29/2020 |

|  |
| --- |
| No: #HF11 |
| Statement: User shall call the python function to display the drone data in the terminal |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The user imports the modify settings package 2. The user calls the display drone function 3. The terminal shall tell the user the drone’s data that is within the settings file |
| Revision History:  Created by Michael Fornito, 10/29/2020 |

## 3.4 Documentation Requirements

* User Manual that explains how to start and stop simulation:
  + PDF is available both online and in printed format
  + The document requires basic reading skills
  + The document requires basic knowledge of using computer
* Detailed design documentation is available for extending the simulation’s capabilities
  + This documentation shall be in PDF format.
  + This documentation shall include a data flow diagram
  + This documentation shall include a UML class diagram
  + This documentation shall include detailed setup instructions for initializing the project
  + This documentation shall include detailed specifications of all configurable variables

## 3.5 Data Requirements

|  |
| --- |
| No: #D1 |
| Statement: A collinear attractive force will be calculated for each actor drone to each sink drone based on the masses of the drones, the current radius of the actor from the sink, and a configurable parameter to control the strength of the force. |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. A force collinear with the direction between a pair of drones is computed with increasing strength as the distance between the drones decreases. |
| Revision History:   1. Created by Tyler Wise, 9/24/20 |

|  |
| --- |
| No: #D2 |
| Statement: A collinear repulsive force will be calculated for each actor drone to every other actor drone based on the masses of the drones, the distance between each pair of drones, and a configurable parameter to control the strength of the force. |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. A force collinear with the direction between a pair of drones is computed with decreasing strength as the distance between the drones increases. |
| Revision History:   1. Created by Tyler Wise, 9/24/20 |

|  |
| --- |
| No: #D3 |
| Statement: The relative position of a given drone to all other drones shall be available at all times. This position shall be updated no less than once every ten seconds. |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The perceived relative position of each drone in a formation is within 10% of true relative position of the drones. |
| Revision History:   1. Created by Tyler Wise, 9/24/20 |

|  |
| --- |
| No: #D4 |
| Statement: The data to control the simulation will be available in the settings.json file. |
| Source: <source of the requirement> |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method:   1. The number of drones in the simulation is reflective of the value in the file. 2. The location of the drones in the simulation is reflective of the values in the file. 3. The distribution of the drones in the simulation is reflective of the value in the file. |
| Revision History:   1. Created by Tyler Wise, 10/28/20 |

## 3.6 Resource Requirements

* Windows 10 platform, 64-bit
* Matlab R2017b - R2018b, 64-bit
* Epic Games Launcher
* Unreal Engine 4.24 or newer
* Visual Studio 2019
* AirSim
* Three sprints
  + Sprint 1 ends 10/1/2020
  + Sprint 2 ends 10/29/2020
  + Sprint 3 ends 11/24/2020

## 3.7 Security Requirements

* Not applicable for this project

## 3.8 Quality Assurance Requirements

* Connection to drones
  + Connection from controller to drones will ideally be continuously maintained, however temporary disruptions may be expected.
  + Messages received from either the drone or controller must be followed by a response, letting the other system know it went through.
  + No response received from the controller to the drones will stop their flight until a new message is received.
  + It is critical that the drones know when a connection is lost. It is encouraged for the controller to know as well to pass that information on to the user. However, since there is no connection, it is unnecessary since neither the user or controller will have control over the drones.
  + When the connection is lost for some time (TBD), the drones will have to initiate a recovery flight back to the controller, either providing better connection, or getting in range for repairs needed.
  + Demonstrated by purposely shutting off the connection after some period of time, and checking the drone logs to ensure it initiated a recovery flight.
* Recovery Flight
  + Extension of losing connection between drones and controller.
  + When the drone notifies itself a connection is lost, it shall use information from the flight to fly straight back to the controller.
  + Information is gathered based on how much of the flight plan was completed (either received from the last controller message, or on board logic to track positioning), and time from last message.
  + The drone, by itself, shall know the location of the controller relative to its location.
  + Demonstrated by sending the Recovery Flight message, and the drone changing its flight plan to go to the controller.
* Downed Drone
  + The drones shall be flying in a predetermined formation, and a drone going down will cause the formation to be adjusted.
  + A lost connection between a single drone and the controller will notify the controller to adjust the remaining drones’ formation.
  + The downed drone can come from either a connection issue or crash, dead battery, or damaged components.
  + In the case of a connection issue, the individual drone will initiate its recovery flight.
  + When the controller does not receive a message from a single drone, it will assume it is lost and send a message to the remaining drone to adjust the formation.
  + Demonstrated by shutting off the connection of a single drone in formation. After a predetermined amount of time (TBD) the controller will send a formation change message to the remaining drones, and each drone should send a response.
* Terminal Input Parameters
  + Out of range values and wrong type of input for calling the python functions:
    - Location (X,Y,Z) where to place the drone, should be integer
    - Number of drones to add, should be integer
    - Remove a drone, should be integer
    - Distribution size, should be integer

# Section 4: Supporting Material

* Reference Projects and Papers
  + APAWSAN: Actor Positioning for Aerial Wireless Sensor and Actor Networks by Mustafa Ilhan Akbas and Damla Turgut, University Of Central Florida
  + Vision Based Formation Control in Unreal Engine -Air Sim by Kaveh Fatian
  + Actor Positioning Based on Molecular Geometry in Aerial Sensor Networks by Mustafa ˙Ilhan Akbas¸ Gurkan Solmaz and Damla Turgut, University of Central Florida