Software Design Document

for

Drone Flight Path Simulator

Version 1.0 approved

Prepared by Alex Alcazar, Francisco Brito, Helen Dam, Alex Gaeta, Alberto Gonzalez, Sergio Maradiaga, Thaddeus Owens, Mychal Salgado, Kevin Tang, Sergio Valadez

DEVCOM Army Research Laboratory

Table of Contents..... <pg 2>
Revision History.....<pg 4>
1. Introduction.....<pg 5>
1.1. Purpose.....<pg 5>
1.2. Document Conventions……..... <pg 5>
1.3. Intended Audience and Reading Suggestions.....<pg 5>
1.4. System Overview..... <pg 5>
2. Design Considerations..... <pg 6>
2.1. Assumptions and dependencies..... <pg 6>
2.2. General Constraints.....<pg 6>
2.3. Goals and Guidelines.....<pg 6>
2.4. Development Methods..... <pg 7>
3. Architectural Strategies.....<pg 8>
4. System Architecture.....<pg 9>
4.1. ..... <pg #>
4.2. ..... <pg #>
5. Policies and Tactics.....<pg #>
5.1. Specific Products Used..... <pg #>
5.2. Requirements traceability.....<pg #>
5.3. Testing the software..... <pg #>
5.4. Engineering tradeoffs..... <pg #>
5.5. Coding guidelines and Conventions..... <pg #>
5.6. Protocol……..... <pg #>
5.7. Choice of algorithms..... <pg #>
5.8. Maintaining the software..... <pg #>
5.9. Interfaces.....<pg #>
5.10. Hierarchical Organization.....<pg #>
5.11. System's deliverables..... <pg #>
5.12. Abstraction.....<pg #>
6. Detailed System Design.....<pg #>
6.1 Name of Module..... <pg #>
6.1.1 Responsibilities..... <pg #>
6.1.2 Constraints..... <pg #>
6.1.3 Composition.....<pg #>
6.1.4 Uses/Interactions.....<pg #>
6.1.5 Resources..... <pg #>
6.1.6 Interface/Exports.....<pg #>
7. Detailed Lower level Component Design
7.1 Name of Class or File..... <pg #>
7.1.1 Classification.....<pg #>
7.1.2 Processing Narrative(PSPEC).....<pg #>
7.1.3 Interface Description.....<pg #>
7.1.4 Processing Detail..... <pg #>
7.1.4.1 Design Class Hierarchy.....<pg #>
7.1.4.2 Restrictions/Limitations.....<pg #>
7.1.4.3 Performance Issues..... <pg #>

7.1.4.4 Design Constraints..... <pg #>
7.1.4.5 Processing Detail For Each Operation..... <pg #>

8. User Interface
8.1. Overview of User Interface..... <pg #>
8.2. Screen Frameworks or Images..... <pg #>
8.3. User Interface Flow Model..... <pg #>
9. Database Design
10. Requirements Validation and Verification..... <pg #>
11. Glossary..... <pg #>
12. References..... <pg #>

Revision History

Name	Date	Reason For Changes	Version
First Draft	12/5/23	Initial Draft of Document	1.0

1. Introduction

1.1 Purpose

The Software Design document will outline the design of the Drone Flight Path Simulator. More to be determined by Spring 2024.

1.2 Document Conventions

- Regular Text - Times New Roman (12 pt font)
- Bold Text - Title and Section Headings
- Numbered lists - Ordered Lists
- Bulleted Lists - Unordered Lists

1.3 Intended Audience and Reading Suggestions

This document is intended to guide D.F.P.S.'s software development developers based on the D.F.P.S. software requirements document. The S.D.D. will act as a baseline for future developers to expand the project if desired.

1.4 System Overview

The D.F.P.S. project will use a web framework to build geographic data supporting a spatial database. Using this database, a user can manipulate spatial indexing, geometry fields, and geographical query sets. We will have front-end map rendering to provide interactive map components. With these functionalities, we can implement a software program that will be able to analyze a given drone flight path, provide feedback, and a mission success rate.

2. Design Considerations

2.1 Assumptions and Dependencies

- Accurate environmental data is available for simulation purposes
    - If not available, allow the User to input any missing data inquiries
- Users possess the requisite knowledge to operate the simulation software
- The environment remains static to match the user's entered data
- Flight path may not change dynamically due to it depending on user data

2.2 General Constraints

- Can't predict how the opposing forces will move while creating the simulation
- Can't rapidly adjust the flight path whenever a detection sensor changes focus
- Limited data from the SUAS sensors
- Limited data for environmental factors

2.3 Goals and Guidelines

- User-Friendly Interface: A friendly UI is crucial for seamless interaction with drone flight planning software, enhancing efficiency, reducing learning curves, especially for small military units, and ensuring accessibility to a wide range of users.

- Ability to Adjust to Terrain & Environment: The software should enable operators to adjust drone flight paths in dynamic military operations swiftly, ensuring mission effectiveness despite unexpected challenges.

- Minimize Detection: The software should incorporate stealthy features for covert drone flight paths, recognize and reduce potential countermeasures, and enhance the drone navigation ability to avoid threats.

- Ensure that the architecture for 3D visualization considers legal requirements and ethical issues.

- Data Protection: security protocols for protecting classified data.
- Drone Battery Life: Optimizing the battery life of a drone allows for extended missions, enabling prolonged surveillance, reconnaissance, and data collection, aligning to utilize sensors effectively.

2.4 Development Methods

Agile development is perfect for dynamic projects with changing demands because it emphasizes prioritizing feedback, working closely with liaisons, and being flexible.
Other development methods will be decided in the next version update.

3. Architectural Strategies

<u>*Preliminary plans but are subject to change</u>

- Database - a database to contain details of a drone detectability such as the base noise level of the drone, the size, etc
- Mapping Environment - Mapbox will be used to give users a 2.5D visualization of their flightpath
- Web-Based Application - Django and map box will be used together to build the application
- Python - A programming language that will be used to build the web application
- JavaScript - programming language that will be used to build the web application
- Graphical User Interface - The Map will take up most of the application view with menus that can be expanded and minimized based on the information the user wants to see
- React/Redux - JS frameworks to help build the application into components and offer better scalability for future endeavors

4. System Architecture

To be determined in Spring 2024

5. Policies and Tactics

5.1 Choice of which specific products used

- Django
- Python
- GitHub
- Visual Studio Code
- MapBox
- GeoDjango

5.2 Plans for ensuring requirements traceability

Our meetings with liaisons and contacts have been recorded to ensure all requirements can be traced to their desired specifications. In addition, all user stories and similar planning steps will be maintained to show how each condition is tied to a user's need.

5.3 Plans for testing the software

- Implement Continuous Implementation using a platform like GitHub Actions to automate testing and ensure code quality before merging into the main branch.
- Because the architecture is relatively new, being built from scratch by this year's team, testing will be done within the following groups (note that it is not open to the public as of December 2023):
    - Internal Portions of the Team
    - Liaisons
    - Contacts of Liaisons
    - Advisors

5.4 Engineering Trade-Offs

- Performance vs Complexity:
    - Goal: High performance and accuracy in drone flight path optimization
    - Approach: Implementing efficient algorithms that optimize flight paths without excessive computational overhead. Some algorithms that have been considered and will be used during testing include A* and Dijkstra's for pathfinding.

- Hardware Limitations:
    - Challenge: Lack of access to drone instrumentation readings.
    - Solution: Create a robust web app with plugin-based architecture to interface with drones through standardized APIs, assuming primary telemetry data. The system should be designed to adapt, implement, and integrate more data as it becomes available.

- Accuracy vs. Computation Time:
    - Priority: High accuracy in flight path optimization
    - Balancing Act: We will consider leveraging cloud computing resources to handle complex computations without compromising response time. Current considerations include AWS Lambda or Azure Functions.

- Safety and Compliance:
    - Consideration: Ensure the flight paths comply with local aviation regulations and safety standards
    - Implementation: Integrate a module in the software that checks and validates flight paths against regulatory constraints.

- Scalability:
    - Objective: Create a scalable solution.
    - Strategy: Implementing a plug-in-based architecture, allowing easy scaling and integration of new features or data sources in the future.

5.5 Coding guidelines and conventions

An essential aspect of our design is ensuring readability since future teams will likely build upon this project. Extensive use of descriptive but concise comments will be implemented throughout the code. All relevant files will also be kept in an organized manner so that each can be found easily. In addition, all general code formatting conventions will be followed in a way that will be consistent across all code. This includes but is not limited to using CamelCase for naming, proper indentation, and descriptive error handling.

5.6 Protocol

To be determined.

5.7 Choice of Algorithms

To be determined.

5.8 Maintaining the software

All code will be claimed via GitHub, and team members' contact information will be provided for future developers and users. Extensive documentation will also be held. Utilizing the latest and most stable versions of software for long-standing support.

5.9 Interfaces

To be determined.

5.10 Hierarchical Organization

To be determined.

5.11 System's Deliverables

To be determined.

5.12 Abstraction

To be determined.

6. Detailed System Design

To be determined in Spring 2024

6.1 Name of Component (Module)

6.1.1 Responsibilities

To be determined in Spring 2024

6.1.2 Constraints

To be determined in Spring 2024

6.1.3 Composition

To be determined in Spring 2024

6.1.4 Uses/Interactions

To be determined in Spring 2024

6.1.5 Resources

To be determined in Spring 2024

6.1.6 Interface/Exports

To be determined in Spring 2024

7. Detailed Lower level Component Design

To be determined in Spring 2024

7.1 Name of Class or File

7.1.1 Classification

To be determined in Spring 2024

7.1.2 Processing Narrative (PSPEC)

To be determined in Spring 2024

7.1.3 Interface Description

To be determined in Spring 2024

7.1.4 Processing Detail

To be determined in Spring 2024

7.1.4.1 Design Class Hierarchy

To be determined in Spring 2024

7.1.4.2 Restrictions/Limitations

To be determined in Spring 2024

7.1.4.3 Performance Issues

To be determined in Spring 2024

7.1.4.4 Design Constraints

To be determined in Spring 2024

7.1.4.5 Processing Detail For Each Operation

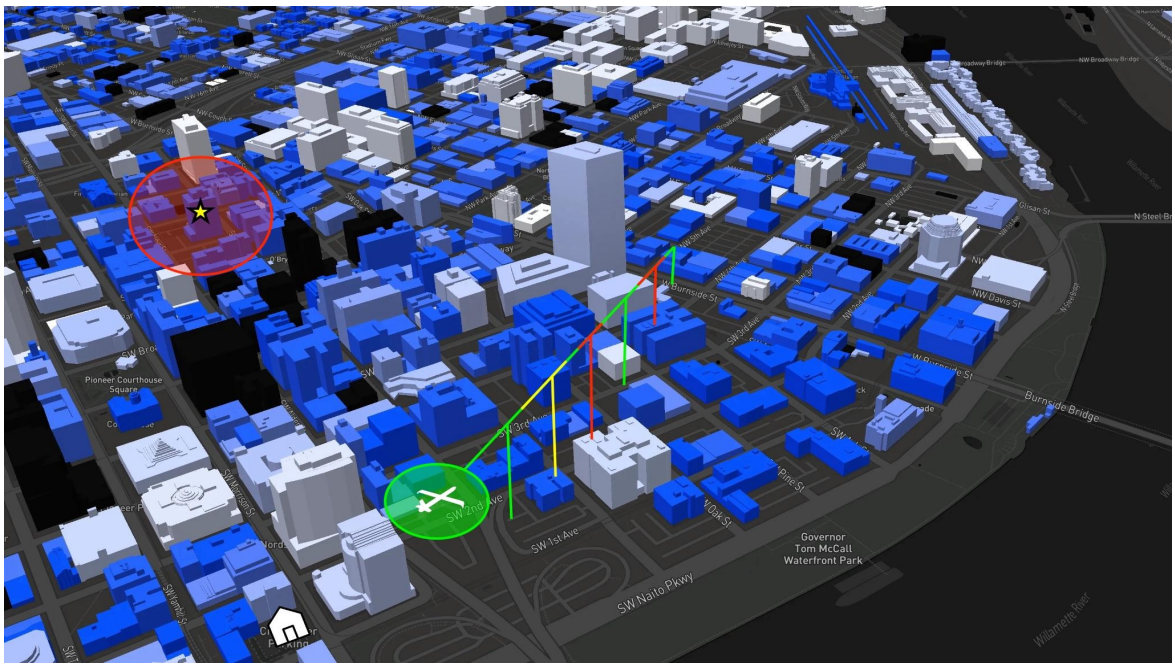
To be determined in Spring 2024

8. User Interface

8.1 Overview of User Interface

To be determined in Spring 2024

8.2 Screen Frameworks or Images



*Subject to change

8.3 User Interface Flow Model

To be determined in Spring 2024

9. Database Design

To be determined in Spring 2024

10. Requirements Validation and Verification

(Not all requirements are set. Subject to change.)

Functional Requirements	Component Modules
D.F.P.S. shall provide a basic 3D geographic environment.	TBD
D.F.P.S. shall allow users to input parameters for the flight path.	TBD
D.F.P.S. shall provide a testable flight path and environment.	TBD
D.F.P.S. shall display an estimated altitude and distance during its flight.	TBD
D.F.P.S. shall provide a selection of drone types, environmental terrains, and weather.	TBD
D.F.P.S. shall detect and mark risk factors within its location using colors and shapes.	TBD
D.F.P.S. shall provide basic sensor characteristics such as audio and visual for drone detection ranges.	TBD
D.F.P.S. shall allow users to save and load flight paths for future uses.	TBD
D.F.P.S. shall provide feedback based on the flight path in the form of a PDF.	TBD
D.F.P.S. shall …	TBD

11. Glossary

- **3D:** Three Dimension
- **A.R.L.:** DEVCOM Army Research Laboratory
- **D.F.P.S.:** Drone Flight Path Simulator
- **O.T.A.:** Over-The-Air
- **PDF:** Portable Document Format
- **SDD:** Software Design Document
- **S.U.A.S.:** Small Unmanned Aircraft System (i.e. Drone)
- **TBD:** To be determined

12. References

Brad Appleton <brad@bradapp.net> http://www.bradapp.net

https://www.cs.purdue.edu/homes/cs307/ExampleDocs/DesignTemplate_Fall08.doc