

Software Requirements Document

Autonomous Cognitive Agent Flying Aircraft in X-Plane

Team Members

Avery Mackey: amackey2023@my.fit.edu

Dylan Ostolaza: dostolaza2024@my.fit.edu

Zachary Reese: zreese2023@my.fit.edu

Brianna Marshall: bmarshall2022@my.fit.edu

Faculty Advisor

Dr. Bhattacharyya: sbhattacharyya@fit.edu

Table of Contents

Autonomous Cognitive Agent Flying Aircraft in X-Plane.....	1
Table of Contents.....	2
1. Introduction.....	3
1.1 Purpose.....	3
1.2 Scope.....	3
1.3 Definitions, Acronyms, and Abbreviations.....	3
1.4 References.....	4
1.5 Overview.....	4
2. Overall Description.....	4
2.1 Product Perspective.....	4
2.2 Product Functions.....	6
2.3 User Characteristics.....	7
2.4 Constraints.....	7
2.5 Assumptions and Dependencies.....	7
3. Specific Requirements.....	7
3.1 External Interfaces.....	7
3.2 Functional Requirements.....	8
3.3 Performance Requirements.....	9
3.4 Design Constraints.....	10
3.5 Software System Attributes.....	10

1. Introduction

1.1 Purpose

The objective of this Software Requirements Specification (SRS) document is to outline all the software requirements for the Autonomous Cognitive Agent Flying Aircraft in X-Plane project. The SRS will provide the development, test, and design teams with a baseline from which to operate as they develop, test, and verify the system.

1.2 Scope

This project extends the existing autonomous aircraft simulation system with the addition of detection and response to engine failures. The system operates within the existing X-Plane and Soar-based environment and can dynamically modify flight routes as required if a failure forces changes.

The system will allow users to create simulated flights and initiate supported failure scenarios. The currently supported equipment failures are:

- Uncharged battery
- Multi-engine failure
- Sensor failures due to aircraft icing

The cognitive agent will detect failure conditions, evaluate the aircraft's remaining capabilities, determine a response, and execute this response. The system will also support multiple failures during a single flight and use X-Plane airport data to identify nearby airports when rerouting is required.

1.3 Definitions, Acronyms, and Abbreviations

- Apt.dat
 - X-Plane airport data file to identify airports close enough to the aircraft to be considered as alternative routes.
- Cognitive Agent
 - A software component based on Soar that assesses the state of an aircraft and provides a response in case of an equipment failure.
- DataRef
 - An X-Plane data variable that can be used to either access or manipulate information from the simulation.
- GUI
 - Graphical User Interface: the visible part of the user interface that allows flight scenario configuration
- MAVLink
 - Micro Air Vehicle Link: a communication protocol that transmits aircraft state and flight status information from the Java application to QGroundControl.
- QGroundControl

- Ground control software used by the current system for transmitting data from a plane and monitoring it.
- Soar
 - The cognitive architecture used for agent reasoning/decision making.
- X-Plane
 - The flight simulator environment is used to represent both the aircraft and its status.
- XPlaneConnect
 - Interface through which the project's software communicates with the X-Plane simulation environment.

1.4 References

[1] IEEE, *IEEE Recommended Practice for Software Requirements Specifications*, IEEE Std 830-1998, reaffirmed 2009.

[2] *Autonomous Cognitive Agent Flying Aircraft in X-Plane*, ACAFAX Project Plan, Florida Institute of Technology.

[3] CCAT_Xplane-QGC, GitHub Repository,
https://github.com/rsen2025-star/CCAT_Xplane-QGC

[4] Laminar Research, *X-Plane 12 System Requirements*,
<https://www.x-plane.com/kb/x-plane-12-system-requirements/>

[5] Soar Cognitive Architecture, *Official Documentation*,
<https://soar.eecs.umich.edu/>

[6] NASA, *XPlaneConnect*, GitHub Repository,
<https://github.com/nasa/XPlaneConnect>

1.5 Overview

Section 2 outlines how the project fits within the existing X-Plane and Soar simulation framework and lists the major functions, users, limitations, and dependencies associated with it. Section 3 defines the project's interface, failure handling, rerouting, performance, and design requirements.

2. Overall Description

2.1 Product Perspective

2.1.1 System Interfaces

- X-Plane simulates the aircraft environment and presents all aircraft state information to the Java application.

- The Java application communicates with X-Plane to transfer relevant information regarding aircraft and failures to the cognitive agent.
- The Soar cognitive agent uses this information and determines appropriate flight responses.
- QGroundControl receives telemetry through MAVLink for real-time flight monitoring.

2.1.2 User Interfaces

- Users can choose a departure airport and a destination airport.
- Users can configure and activate supported equipment failures.
- The GUI displays the flight route at its current location.
- The GUI displays the currently active equipment failures.
- The GUI displays the cognitive agent's chosen response.
- The GUI indicates when the flight route has been modified.

2.1.3 Hardware Interfaces

- The system runs on a single computer that can handle both X-Plane 12 and the other required software listed in 2.1.4.
- The computer must support X-Plane and the required software components simultaneously.
- No actual aircraft, flight control computer, or avionics equipment is needed.

2.1.4 Software Interfaces

- XPlaneConnect enables interaction between the Java program and the X-Plane flight simulator.
- DataRefs allow the Java application to retrieve an aircraft's equipment and state information from within X-Plane.
- Soar provides the Cognitive Agent with its reasoning and decision-making capabilities.
- X-Plane's apt.dat file contains all of the airports that are used for the nearby airport lookup process.

2.1.5 Communication Interfaces

- X-Plane and the Java application exchange simulation data via XPlaneConnect and DataRefs.
- The Java application exchanges aircraft state, equipment failure, and decision information with the Soar cognitive agent.
- Flight status and aircraft state information are sent to QGroundControl via MAVLink for observation.

2.1.6 Memory

The system maintains all information about an aircraft's status, active failures, flight path, and the cognitive agent's state during system operation. No additional long-term memory has been identified.

2.1.7 Operations

Normal operations begin when the user configures an environment to simulate a flight along with supported failure modes. The route is determined as part of this configuration process and provided to both the simulation and the cognitive agent.

Supported failures can be initiated during normal flight operations. In these instances, the cognitive agent will evaluate whether a reroute is necessary. To assist in determining possible routes, the system uses X-Plane's airport database to identify the nearest airports. This information is then made available to the cognitive agent so it can modify its route accordingly.

2.1.8 Site Adaptation Requirements

There are no site-specific adaptability requirements for this project, as this system will run in conjunction with the current project simulation environment. There are potential installation and machine-specific configurations for X-Plane, XPlaneConnect, Java, Soar, and any additional existing project software.

2.2 Product Functions

2.2.1 Flight Setup

- The system enables users to configure a simulated flight by specifying the departure airport and destination airports and choosing one or more supported equipment failures during the flight operation.

2.2.2 Initial Flight Route

- The system obtains an initial route from the departure airport to the destination airport to be used by the cognitive agent and simulation.

2.2.3 Equipment Failure Detection

- The cognitive agent monitors the simulated aircraft state to identify any supported equipment failures. Upon detecting a failure, the agent updates its representation of the aircraft's remaining capabilities.

2.2.4 Equipment Failure Response

- After detecting an equipment failure, the cognitive agent determines a response based on the aircraft's remaining capabilities, which is then communicated to X-Plane.

2.2.5 Dynamic Flight Route Modification

- When a failure requires a change to the current route, the system obtains nearby airports via X-Plane's apt.dat data. The cognitive agent uses the returned information to determine a new flight route.

2.2.6 Multiple Failure Handling

- The cognitive agent evaluates the combined effect of all active failures when determining the aircraft's response.

2.2.7 Flight Status Display

- The system displays to the user the current flight route, initiated equipment failures, detected equipment failures, the cognitive agent's response, and flight route modifications.

2.3 User Characteristics

The primary users of the system are individuals operating and evaluating the autonomous aircraft simulation, including project team members, developers, and researchers. These users are expected to have a general understanding of simulated flight operations and the project environment, including X-Plane and project-specific software. Although users do not need detailed knowledge of the internal Soar architecture to operate the system, familiarity with the supported equipment failure scenarios may assist in their understanding of the system's behavior.

2.4 Constraints

- The system is limited to equipment failures that can be represented within the X-Plane simulation environment.
- The system is limited by the capabilities of the selected aircraft model.
- The system must operate within a pre-established simulation environment.

2.5 Assumptions and Dependencies

Assumptions:

- X-Plane and the existing project software are functioning during operation.
- The selected aircraft model provides the information needed to simulate the supported equipment failure scenarios.

Dependencies:

- X-Plane, Soar, the Java application, and the existing XPlaneConnect/DataRef integration.
- X-Plane airport data for nearby-airport lookup.

3. Specific Requirements

3.1 External Interfaces

3.1.1 User Interfaces

- The system shall:
 - Provide controls for selecting a departure airport and destination airport.
 - Provide controls to configure and initiate supported equipment failures.
 - Display the current flight route.
 - Display active equipment failures.
 - Display the cognitive agent's selected response.
 - Visually indicate when the flight route has changed.

3.1.2 Hardware Interfaces

- The system shall:
 - Operate on a computer capable of running both X-Plane and the supporting project software.
 - Meet or exceed the minimum hardware requirements specified for X-Plane 12

3.1.3 Software Interfaces

- The system shall:
 - Interface with the Soar cognitive agent to provide aircraft state information, failure information, and receive agent decisions.
 - Interface with the Soar cognitive agent to send aircraft and failure information and receive agent decisions.
 - Access X-Plane airport data from apt.dat for dynamic route modification.
 - Maintain compatibility with the existing XPlaneConnect/DataRef integration.

3.1.4 Communication Interfaces

- The system shall:
 - Support communication between X-Plane, the Java application, and the Soar cognitive agent.
 - Support the existing communication with QGroundControl where required by the current system.

3.2 Functional Requirements

3.2.1 Flight Setup

- The system shall:
 - Allow users to select a departure airport.
 - Allow users to select a destination airport.
 - Allow users to configure one or more supported equipment failures for the flight.
 - Reflect each initiated equipment failure in the X-Plane aircraft state.

3.2.2 Initial Flight Route

- The system shall:
 - Obtain an initial flight route for the selected departure and destination.
 - Provide the initial route to the cognitive agent and X-Plane simulation.

3.2.3 Equipment Failure Detection

- The cognitive agent shall:
 - Detect the supported equipment failures after they occur.
 - Uncharged battery
 - Multi-engine failure
 - Sensor failures due to aircraft icing
 - Identify the active supported failure condition.
 - Update its internal representation of the aircraft's remaining capabilities after a failure is detected.

3.2.4 Equipment Failure Response

- The cognitive agent shall:
 - Determine a response to a detected equipment failure.
 - Consider the type of detected failure when determining the response.
 - Consider the aircraft's remaining capabilities when determining the response.
 - Reevaluate the response if an additional supported failure occurs.
- The system shall communicate response actions to X-Plane.

3.2.5 Dynamic Flight Route Modification

- The cognitive agent shall:
 - Determine when to modify the current flight route in response to detected failures.
 - Use nearby airport information to determine a revised flight route.
- The system shall:
 - Access airport information from X-Plane's apt.dat file.
 - Select the N nearest airports to the aircraft when it needs to be routed.
 - Deliver the airport information results to the cognitive agent.
 - Present the new routing path to X-Plane.

3.2.6 Multiple Failure Handling

- The system shall:
 - Support multiple equipment failures during a single flight.
 - Display all active supported failures to the user.
- The cognitive agent shall:
 - Detect all active supported failures.
 - Consider the combined effect of active failures on the aircraft's remaining capabilities.
 - Determine a response based on the combined failure state.

3.2.7 Flight Status Display

- The system shall:
 - Display the current flight route.
 - Display active equipment failures.
 - Display the cognitive agent's selected response.
 - Denote when the flight route changes.

3.3 Performance Requirements

- The system shall:
 - Detect equipment failure within **5 seconds** after a supported piece of equipment has failed.
 - Correctly identify supported equipment failure conditions in at least **70%** of test cases.
 - Generate an alternate flight route within **5 seconds** after determining that a route change is required.

- Update the displayed route, failure, and agent-response information within **5 seconds** of a change.
- Support at least **3 simultaneous equipment failures** during a single flight scenario.
- Identify the N closest airports within **5 seconds** after nearby-airport information is requested.
- The cognitive agent shall:
 - Determine a response to a detected equipment failure within **5 seconds**.
 - Select the appropriate response for a detected equipment failure in at least **70%** of test cases

3.4 Design Constraints

The system shall be compatible with the current simulation environment as an extension of the existing software.

- The system shall:
 - Operate within the existing X-Plane simulation environment.
 - Use the Soar cognitive architecture for agent decision making.
 - Merge with the existing project codebase rather than replacing the current system architecture.

3.5 Software System Attributes

3.5.1 Usability

- The system shall:
 - Provide flight routes, equipment failures, cognitive-agent responses, and route modifications in a clear and understandable format.
 - Enable the users to operate the GUI without requiring detailed knowledge of the internal Soar architecture.

3.5.2 Reliability

- The system shall:
 - Maintain consistent aircraft state and failure information during normal operation.
 - Continue to operate when multiple equipment failures are active.

3.5.3 Maintainability

- The system shall:
 - Enable additional equipment failure scenarios to be added without replacing the entire existing architecture.
 - Enable modification or expansion of cognitive agent response logic for new failure scenarios that are introduced.