Student Specifications Document Template Guide

Tanner Chase

Christian Hall

Anthony Wood

November 1, 2025

Signature Page

Signature	*Signature*	*Signature*
Date and email	*Date and email*	*Date and email*

Revision History

Revision	Description	Author	Date	Approval
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

Contents

Specifications

1 SCOPE

(a) **General:**

(b) **Acronyms:**

2 APPLICABLE DOCUMENTS

The following documents shown shall form part of the specifications for this project. In the event of a conflict between requirements, priority shall first go to the contract, second to this document, and lastly to these reference documents.

(a) **Government Documents** *This is where to put MIL-Specs, MIL-STDs, NASA specs and so forth. Be sure to include the revision level and date.*

(b) **Industry Documents** *This is where to put ANSI, ASTM, ASME, IEEE, Company specifications and so forth. Both this section and government documents can be divided up into logical subcategories.*

3 STAKEHOLDER REQUIREMENTS

4 ENGINEERING REQUIREMENTS

4.1 System Requirements

Synopsis of System

4.1.1 Processing

Processing Pipeline

4.1.2 Built in HW Capabilities

Processing Pipeline

4.1.3 HW Interfaces

Processing Pipeline

4.1.4 HAL

Processing Pipeline

4.1.5 Power Management

Processing Pipeline

4.1.6 Functional Requirements

(a) Requirement 1

4.1.6.1 Submodules

(a) Processing Core

(b) Processing Architecture

(c) Hardware Platform

(d) Configuration Tool

4.1.7 Inputs

(a) *Input 1*

Stuff

4.1.8 Outputs

(a) *Output 1*

Stuff

4.1.9 Internal Interfaces

(a) *Blah*

Stuff

4.1.10 Non-Functional Requirements

Energy

(a) Energy Requirement 1

Environments

(a) Environment Requirement 1

Safety

(a) Safety Requirement 1

Structure

(a) Structure Requirement 1

Standards and Regulations

(a) Standards and Regulations Requirement 1

4.1.11 Submodules

The System is comprised of four functional modules.

(a) Planner

(b) Controller

(c) Navigation

(d) Logger

4.2 Planner

The purpose of the Planner is to generate a series of realizable states for the UAV. The planner allows for both autonomous and pilot-directed operation. Inputs from both autonomous and pilot-directed modes are evaluated and potentially adjusted to fit safety and feasibility constraints. The inputs are evaluated based on the current UAV state and the current tolerances from configuration parameters.

4.2.1 Functional Requirements

(a) Generate feasible waypoint trajectories based on input commands and UAV state.

4.2.2 Inputs

(a) *Waypoints and configuration parameters*

Stuff

4.2.3 Outputs

(a) *Planned waypoints*

Stuff

4.2.4 Internal Interfaces

(a) *Blah*

Stuff

4.2.5 Non-Functional Requirements

Energy

(a) The Planner shall operate within the power budget allocated for the onboard processor.

Environments

(a) The Planner shall tolerate expected operating temperatures and vibration levels.

Safety

(a) The Planner shall not issue unsafe actuator commands under fault conditions.

Structure

(a) The Planner shall expose well-defined APIs for its submodules.

Standards and Regulations

(a) The Planner shall follow applicable coding and safety guidelines.

4.2.6 Submodules

The Planner is comprised of three submodules. Each has aids in the overall functionality of the Planner.

(a) Waypoint Planner

(b) RC Mixer

(c) State Select

4.3 Waypoint Planner

The Waypoint Planner submodule is responsible for generating feasible waypoint trajectories based on input commands and the current UAV state. It processes both autonomous and pilot-directed inputs, ensuring that the generated waypoints adhere to safety and feasibility constraints.

4.3.1 Functional Requirements

(a) FR 1

4.3.2 Inputs

(a) Waypoints and configuration parameters as defined in XXX:location

4.3.3 Outputs

(a) Planned waypoints as defined in XXX:location

4.3.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Ensure generated waypoints adhere to safety constraints.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.4 RC Mixer

Crap

4.4.1 Functional Requirements

(a) FR 1

4.4.2 Inputs

(a) Item as defined in XXX:location

4.4.3 Outputs

(a) Item as defined in XXX:location

4.4.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Ensure generated waypoints adhere to safety constraints.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.5 State Select

Crap

4.5.1 Functional Requirements

(a) FR 1

4.5.2 Inputs

(a) Item as defined in XXX:location

4.5.3 Outputs

(a) Item as defined in XXX:location

4.5.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Ensure generated waypoints adhere to safety constraints.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.6 Controller

Compute actuator setpoints from planned trajectories and state estimates.

4.6.1 Functional Requirements

(a) Implement required control algorithms and scheduling.

4.6.2 Inputs

(a) *In 1*

Descr.

4.6.3 Outputs

(a) *Out 1*

Descr.

4.6.4 Internal Interfaces

(a) *Int 1*

Descr.

4.6.5 Non-Functional Requirements

Energy

(a) Execute within the control loop CPU and power budgets.

Environments

(a) Maintain stability across environmental conditions and sensor noise.

Safety

(a) Ensure bounded outputs and safe behavior under saturation and faults.

Structure

(a) Modular controller design with clear scheduling and interfaces.

Standards and Regulations

(a) Follow applicable control and software standards.

4.6.6 Submodules

The Controller is comprised of four submodules that work together to translate planned trajectories into actuator commands.

(a) Preprocessor

(b) PVA Controller

(c) ATL Controller

(d) Control Distributor

4.7 Preprocessor

The Preprocessor submodule is responsible for preparing and conditioning inputs for the control loops, including filtering and validation of sensor data before it is used by the controllers.

4.7.1 Functional Requirements

(a) Provide filtered and validated sensor data to controllers.

4.7.2 Inputs

(a) Raw sensor measurements as defined in XXX:location

4.7.3 Outputs

(a) Filtered sensor streams as defined in XXX:location

4.7.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Filtering shall never mask critical fault indicators.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.8 PVA Controller

The PVA (Position/Velocity/Acceleration) Controller submodule computes control setpoints to track planned trajectories and maintain the UAV along the desired path.

4.8.1 Functional Requirements

(a) Maintain tracking error within specified bounds.

4.8.2 Inputs

(a) Waypoints and state estimates as defined in XXX:location

4.8.3 Outputs

(a) Actuator-level setpoints as defined in XXX:location

4.8.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Ensure bounded errors and safe fallback when estimates are invalid.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.9 ATL Controller

The ATL (Axis/Attitude/Torque-Level) Controller submodule translates higher-level commands into low-level actuator demands, respecting actuator limits and constraints.

4.9.1 Functional Requirements

(a) Provide actuator commands that respect actuator limits.

4.9.2 Inputs

(a) Desired attitude and torque references as defined in XXX:location

4.9.3 Outputs

(a) Low-level actuator commands as defined in XXX:location

4.9.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Enforce actuator limits and safe modes under faults.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.10 Control Distributor

The Control Distributor submodule routes control outputs to the correct actuator channels, mapping logical control channels to physical hardware outputs.

4.10.1 Functional Requirements

(a) Map logical control channels to hardware outputs.

4.10.2 Inputs

(a) Controller output commands as defined in XXX:location

4.10.3 Outputs

(a) Hardware actuator signals as defined in XXX:location

4.10.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Prevent misrouting; provide diagnostics and traceability.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.11 Navigation

Provide accurate state estimation and sensor selection to support planning and control by running estimators, selecting appropriate sensors, and publishing state estimates.

4.11.1 Functional Requirements

(a) Provide timely, bounded-uncertainty state estimates.

4.11.2 Inputs

(a) *In 1*

Descr.

4.11.3 Outputs

(a) *Out 1*

Descr.

4.11.4 Internal Interfaces

(a) *Int 1*

Descr.

4.11.5 Non-Functional Requirements

Energy

(a) Complete estimation within allocated compute budget.

Environments

(a) Maintain accuracy across sensor noise and environmental changes.

Safety

(a) Provide validity flags and fail-safe estimates upon degraded inputs.

Structure

(a) Provide uncertainty metrics and timestamps with each estimate.

Standards and Regulations

(a) Follow applicable estimation and data handling standards.

4.11.6 Submodules

The Navigation module is comprised of two submodules that work together to provide accurate state estimation from available sensor inputs.

(a) Sensor Selector

(b) Estimator

4.12 Sensor Selector

The Sensor Selector submodule evaluates available sensor inputs and selects the best sensors for the current conditions, ensuring robust state estimation even when some sensors fail or degrade.

4.12.1 Functional Requirements

(a) Provide prioritized sensor streams and fallback choices.

4.12.2 Inputs

(a) All available sensor feeds and health indicators as defined in XXX:location

4.12.3 Outputs

(a) Selected sensor feeds for estimator as defined in XXX:location

4.12.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Ensure safe fallbacks when preferred sensors are unavailable.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.13 Estimator

The Estimator submodule fuses selected sensor inputs using sensor fusion algorithms to produce the best estimate of the system state (position, velocity, attitude, etc.).

4.13.1 Functional Requirements

(a) Provide pose, velocity and other required state variables with uncertainty metrics.

4.13.2 Inputs

(a) Selected sensor streams and configuration as defined in XXX:location

4.13.3 Outputs

(a) State estimates with timestamps as defined in XXX:location

4.13.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Provide safe outputs when inputs are inconsistent or stale.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.14 Logger

Record mission data and provide mechanisms for transmission or storage by collecting telemetry, writing to storage, and/or sending data via communications links.

4.14.1 Functional Requirements

(a) Preserve critical telemetry and provide retrieval interfaces.

4.14.2 Inputs

(a) *In 1*

Descr.

4.14.3 Outputs

(a) *Out 1*

Descr.

4.14.4 Internal Interfaces

(a) *Int 1*

Descr.

4.14.5 Non-Functional Requirements

Energy

(a) Logging shall not exceed allocated power/CPU budgets.

Environments

(a) Maintain logging integrity across operating conditions.

Safety

(a) Logging should not interfere with real-time control operations.

Structure

(a) Provide schemas and retention policies for stored data.

Standards and Regulations

(a) Follow applicable data retention and privacy regulations.

4.14.6 Submodules

The Logger is comprised of two submodules that handle data persistence and transmission for mission telemetry and diagnostic information.

(a) Storage

(b) Transmitt

4.15 Storage

The Storage submodule is responsible for persisting telemetry and logs to onboard non-volatile memory, ensuring data is retained for later retrieval and analysis.

4.15.1 Functional Requirements

(a) Support retrieval by timestamp and event markers.

4.15.2 Inputs

(a) Telemetry streams to be persisted, storage configuration as defined in XXX:location

4.15.3 Outputs

(a) Stored log files or records retrievable by time/index as defined in XXX:location

4.15.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Protect critical logs and prevent data loss on faults.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

4.16 Transmitt

The Transmitt submodule handles transmission of selected telemetry data to ground stations or other external systems via available communications links.

4.16.1 Functional Requirements

(a) Provide telemetry prioritization and retransmission.

4.16.2 Inputs

(a) Selected telemetry streams and transmission policy as defined in XXX:location

4.16.3 Outputs

(a) Telemetry packets transmitted via communications link as defined in XXX:location

4.16.4 Non-Functional Requirements

Energy

(a) Execute within allocated CPU and power budgets.

Environments

(a) Maintain performance across expected environmental conditions.

Safety

(a) Prioritize safety/health telemetry under degraded links.

Structure

(a) Modular design with clear interfaces.

Standards and Regulations

(a) Follow applicable software development standards.

5 VERIFICATION OF REQUIREMENTS

Possible verification methods include:

1. Inspection:
   Inspection is a method of verification consisting of investigation, without the use of special laboratory appliances or procedures, to determine compliance with requirements. Inspection is generally nondestructive and includes (but is not limited to) visual examination, manipulation, gauging, and measurement.

2. Demonstration:
   Demonstration is a method of verification that is limited to readily observable functional operation to determine compliance with requirements. This method shall not require the use of special equipment or sophisticated instrumentation.

3. Analysis:
   Analysis is a method of verification, taking the form of the processing of accumulated results and conclusions, intended to provide proof that verification of a requirement has been accomplished. The analytical results may be based on engineering study, compilation or interpretation of existing information, similarity to previously verified requirements, or derived from lower level examinations, tests, demonstrations, or analyses.

4. Direct Test:

   Test is a method of verification that employs technical means, including (but not limited to) the evaluation of functional characteristics by use of special equipment or instrumentation, simulation techniques, and the application of established principles and procedures to determine compliance with requirements.

5.1 Verify Coverage of Stakeholder Requirements

Paragraph Number	Test Type	Tester's Name	Pass/Fail	Date