Sharper Specs for Smarter Drones: Formalising Requirements with FRET

Oisín Sheridan Leandro Buss Becker Marie Farrell Matt Luckcuck Rosemary Monahan

Department of Computer Science, Maynooth University/Hamilton Institute, Maynooth, Ireland

Automation and Systems Department, Federal University of Santa Catarina, Florianópolis, Brazil

Department of Computer Science, The University of Manchester, Manchester, UK

School of Computer Science, University of Nottingham, Nottingham, UK

[illegible]

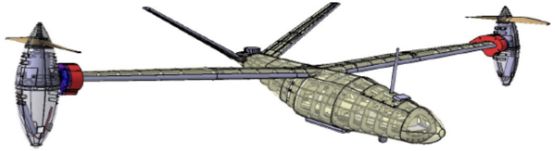
Introduction

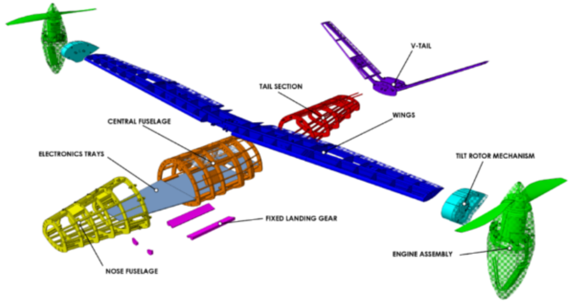
Overview

- We describe the process of formalising the natural-language requirements for a tilt-rotor drone using the Formal Requirements Elicitation Tool (FRET).
- This requirements set evolved over four distinct versions as new information was elicited and incorporated into the FRETish specification.
- Our two concrete outputs are the formalised requirement set, which we will use in our ongoing development and verification of ProVANT; and metrics about the requirements.
- We present guidance for requirements elicitation and formalisation with FRET. We highlight situations where it was difficult to formalise these requirements and describe potential improvements to FRET to address these difficulties.

Case Study – ProVANT Emergentia Tilt-Rotor Drone

Case Study – Tilt-Rotor Drone





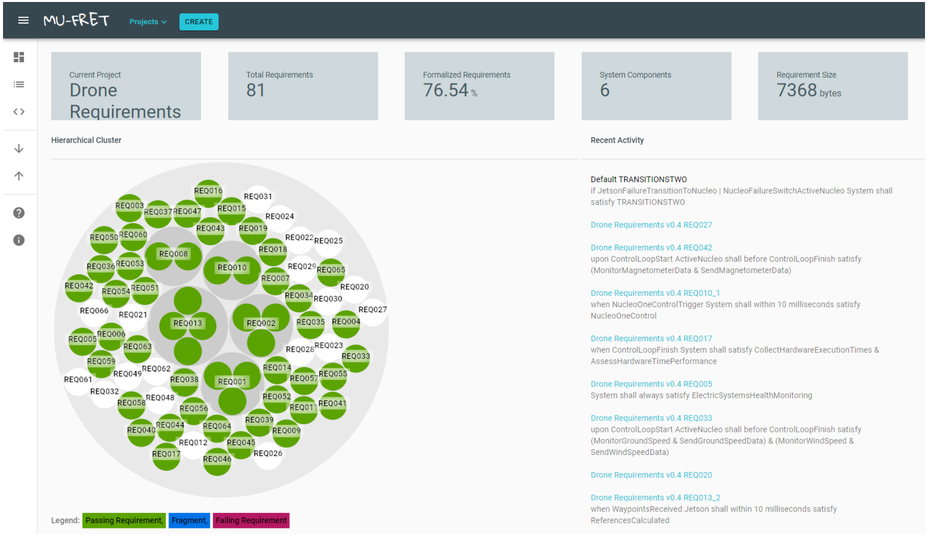
- At the latter stage of development under the ProVANT Emergantia project
- The project is a collaboration among two Brazilian universities, Federal University of Minas Gerais (UFMG) and Federal University of Santa Catarina (UFSC), along with the University of Seville, Spain.

Case Study – Tilt-Rotor Drone

- The drone can perform hovering and Vertical Take-off and Landing (VTOL) manoeuvres, as well as cruise flight as a fixed-wing aircraft.
- The architecture for the Drone's computing system comprises four main components:
  - Raspberry Pi: Gathers sensor data and communicates with the Ground Control Station.
  - Jetson: Processes sensor data and runs the control algorithm.
  - Nucleos: the active nucleo interfaces with the drone's actuators and some sensors. Can also run a backup control algorithm in the case of a failure. There are two nucleos for reliability.
- The set of requirements for the ProVANT Emergentia drone includes aspects related to:
  1. operation features present during simulations and during real executions
  2. remote monitoring configurations
  3. timing constraints associated with the control loop
  4. operation modes under failure conditions

Formalisation with FRET

The Formal Requirements Elicitation Tool (FRET)



Current Project
Drone Requirements

Total Requirements
81

Formalized Requirements
76.54 %

System Components
6

Requirement Size
7368 bytes

Hierarchical Cluster

Legend: 🟩 Passing Requirement, 🟦 Fragment, 🟥 Failing Requirement

Recent Activity

Default TRANSITIONSTWO
if JetsonFailureTransitionToNucleo | NucleoFailureSwitchActiveNucleo System shall satisfy TRANSITIONSTWO

Drone Requirements v0.4 REQ027

Drone Requirements v0.4 REQ042
upon ControlLoopStart ActiveNucleo shall before ControlLoopFinish satisfy (MonitorMagnetometerData & SendMagnetometerData)

Drone Requirements v0.4 REQ010_1
when NucleoOneControlTrigger System shall within 10 milliseconds satisfy NucleoOneControl

Drone Requirements v0.4 REQ017
when ControlLoopFinish System shall satisfy CollectHardwareExecutionTimes & AssessHardwareTimePerformance

Drone Requirements v0.4 REQ005
System shall always satisfy ElectricSystemsHealthMonitoring

Drone Requirements v0.4 REQ033
upon ControlLoopStart ActiveNucleo shall before ControlLoopFinish satisfy (MonitorGroundSpeed & SendGroundSpeedData) & (MonitorWindSpeed & SendWindSpeedData)

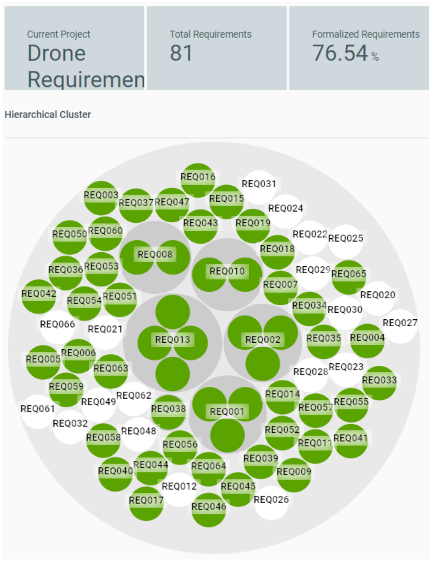
Drone Requirements v0.4 REQ020

Drone Requirements v0.4 REQ013_2
when WaypointsReceived Jetson shall within 10 milliseconds satisfy ReferencesCalculated

The Formal Requirements Elicitation Tool (FRET)

FRET

- An open source tool for requirements engineering developed by NASA
- Requirements are written in a structured natural-language called FRETish
- FRET provides automated translations from FRETish to CoCoSpec contracts, which can be verified with the Kind2 model checker, and Copilot runtime monitors
- Formalised requirements are indicated in green, while those in white have not been formalised

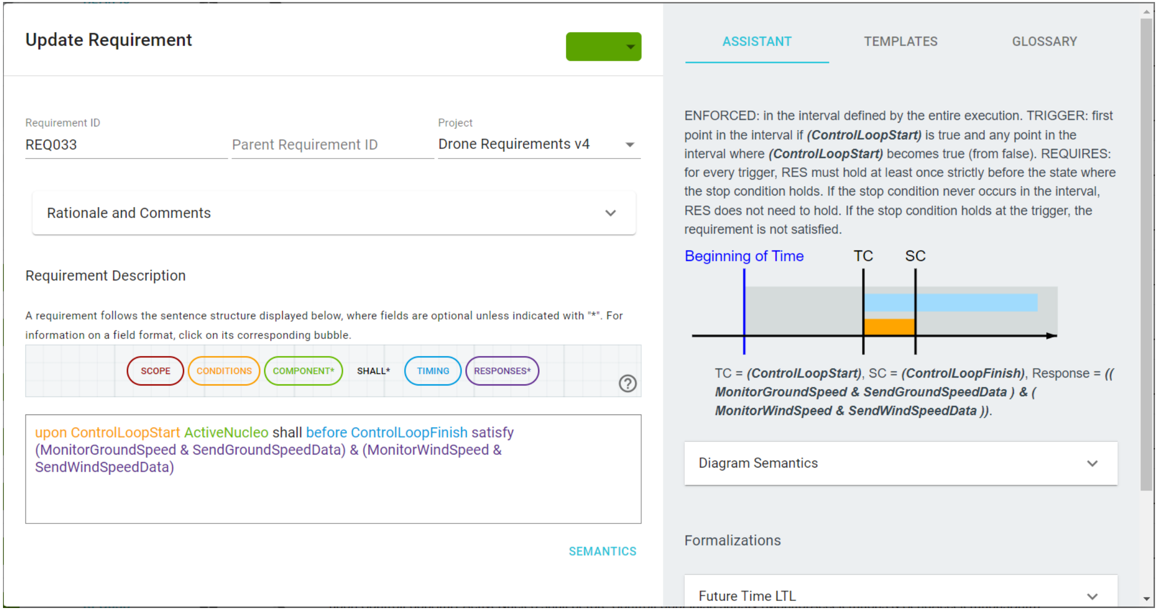


Current Project
Drone Requiremen

Total Requirements
81

Formalized Requirements
76.54 %

Hierarchical Cluster



Update Requirement

ASSISTANT TEMPLATES GLOSSARY

Requirement ID
REQ033

Parent Requirement ID

Project
Drone Requirements v4

Rationale and Comments

Requirement Description

A requirement follows the sentence structure displayed below, where fields are optional unless indicated with "*". For information on a field format, click on its corresponding bubble.

SCOPE CONDITIONS COMPONENT* SHALL* TIMING RESPONSES*

upon ControlLoopStart ActiveNucleo shall before ControlLoopFinish satisfy (MonitorGroundSpeed & SendGroundSpeedData) & (MonitorWindSpeed & SendWindSpeedData)

SEMANTICS

ENFORCED: in the interval defined by the entire execution. TRIGGER: first point in the interval if ***(ControlLoopStart)*** is true and any point in the interval where ***(ControlLoopStart)*** becomes true (from false). REQUIRES: for every trigger, RES must hold at least once strictly before the state where the stop condition holds. If the stop condition never occurs in the interval, RES does not need to hold. If the stop condition holds at the trigger, the requirement is not satisfied.

TC = ***(ControlLoopStart)***, SC = ***(ControlLoopFinish)***, Response = ***((MonitorGroundSpeed & SendGroundSpeedData) & (MonitorWindSpeed & SendWindSpeedData))***.

Diagram Semantics

Formalizations

Future Time LTL