A Novel Approach to CubeSat Flight Software Development Using Robot Operating System (ROS)

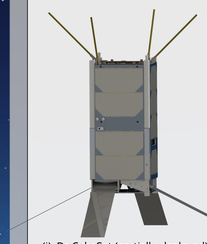
SSC20-P3-18

Samuel Buckner, Carlos Carrasquillo, Marcus Elosegui and Riccardo Bevilacqua
University of Florida Department of Mechanical & Aerospace Engineering
939 Center Dr, Gainesville, FL 32611

Meet the Demo Missions

Drag De-Orbit Device (D3)

The D3 CubeSat mission aims to show how aerodynamic drag can be used as a means for orbital maneuvering, collision avoidance and de-orbit location (latitude/longitude) targeting, in addition to maximizing orbital decay to support Space Situational Awareness (SSA) efforts. This is facilitated through the deployment of four booms that can be deployed and retracted to any intermediate length, allowing modulation of the aerodynamic drag experienced by the satellite while in low Earth orbit (LEO). Software development is targeting completion by the end of 2020, with a final launch date targeting no earlier than **October 2021.**



(i). D3 CubeSat (partially-deployed)

(ii). D3 CubeSat (fully-deployed)

Mission Concept of Operations:

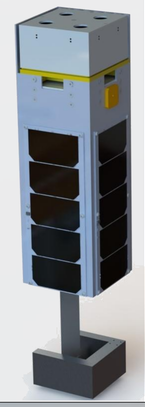
1. Deploy CubeSat from ISS payload module
2. Initialize power and avionics systems
3. Command magnetorquer with B-dot detumbling algorithm for 2-axis attitude stabilization
4. Sit idle at maximum boom deployment to ensure maximized orbital decay
5. Just before targeting controllability is no longer feasible (approx. 1-2 weeks before de-orbit):
   1. Upload initial guidance profile with desired vehicle state and ballistic coefficients
   2. Engage guidance tracker with LQR (linear-quadratic-regulator) control to slightly modify ballistic coefficients and keep the vehicle on the desired guidance profile
6. Periodically regenerate and uplink new guidance profiles for accurate targeting to de-orbit

Omar, Sanny R., and Riccardo Bevilacqua. "Hardware and GNC solutions for controlled spacecraft re-entry using aerodynamic drag." Acta Astronautica 159 (2019): 49-64.

Passive Thermal Coating Observatory Operating in Low earth orbit (PATCOOL)

The PATCOOL CubeSat mission is a testbed for the performance of experimental cryogenic selective surface samples in LEO (low-Earth orbit). To accurately portray the Sun's affects on the samples, the vehicle is kept zenith pointing using a custom ADCS (attitude determination and control system) combining a single D3 boom (with a tip-mass) and a magnetorquer for passive 3-axis stabilization. Deadlines are currently the same as those listed above for D3.

The PATCOOL software requirements can be considered a subset of D3. For example, the boom actuator can be modified for use with one boom, the guidance tracker is unnecessary and hardware interface nodes can be interchanged, among other possibilities. This refurbishment of the D3 flight software will accelerate the PATCOOL software development timeline and prove out the overall software robustness to different mission needs and operating environments.



Ojeda, Carlos, et al. "Passive Thermal Coating Observatory Operating in Low-Earth Orbit (PATCOOL)--Cubesat Design to Test Passive Thermal Coatings in Space." 5th IAA Conference on University Satellite Missions and CubeSat Workshop (2020)

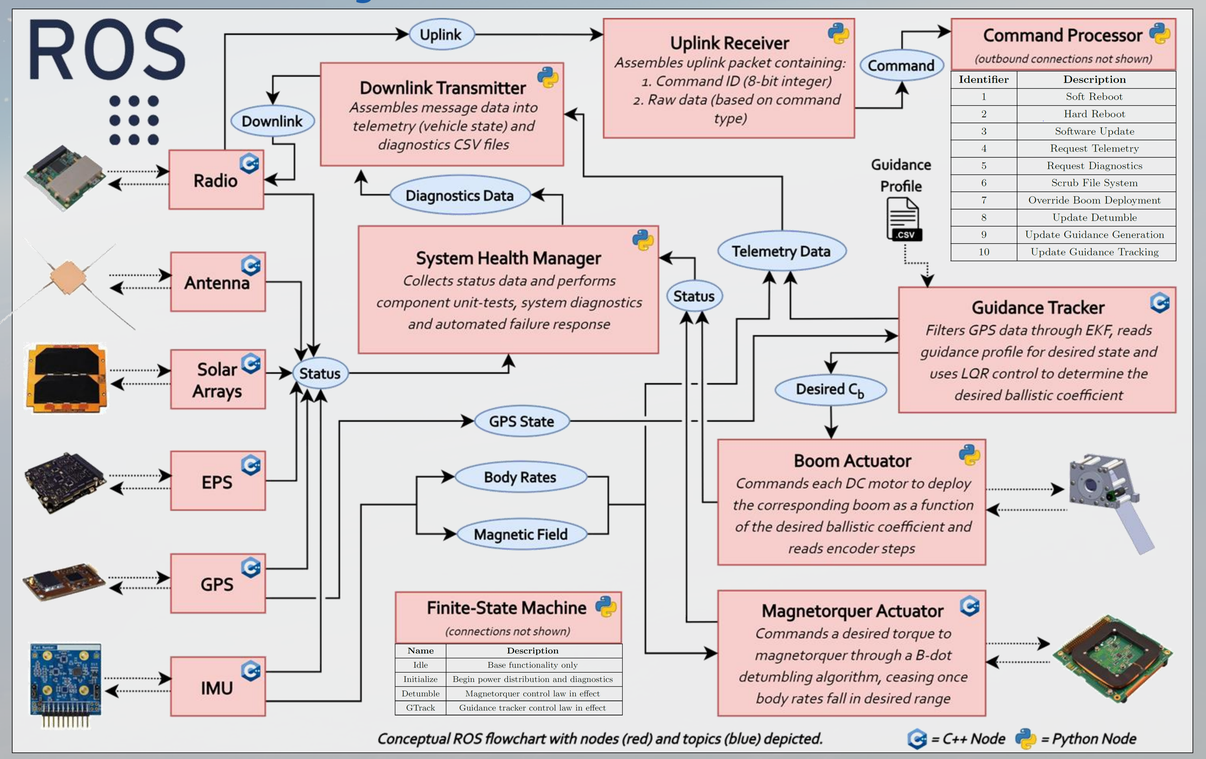
Overview

The large-scale development and deployment of small satellite systems in the modern aerospace sector motivates the parallel development of new flight software and avionics systems to meet the demands of modern spaceflight operations. **Robot Operating System (ROS)** provides an open-sourced, modular and efficient multiprocessing software framework that can be adapted for space-faring purposes in addition to the standard robotics operations that it is intended for.

At the University of Florida's Advanced Autonomous Multiple Spacecraft (ADAMUS) Laboratory, this framework, serving as the first flight-ready implementation of ROS in spacecraft flight software architecture, will be used in two upcoming CubeSat missions: (1) **The Drag De-Orbit Device (D3)** mission and (2) **The PAssive Thermal Coating Observatory Operating in Low earth orbit (PATCOOL)** mission. These missions will serve to validate the reusability of this software and the core functionality contained within, supporting tasks such as data link transmission, command processing, GNC (guidance, navigation & control), avionics system health diagnostics and soft/hard reboot instantiation. The software will operate on the 32-bit ARM processor aboard a BeagleBone Black flight computer.

The structure of ROS can be decomposed into a set of software modules (**nodes**) communicating over a language-agnostic data transfer protocol (**messages**) using a publishers and subscriber routine (**topics**). Characteristics of ROS include modularity (individual nodes can be easily swapped), reusability (functionality can be tweaked supporting multiple mission configurations) and multi-lingualism (hybrid Python/C++ code-base). ROS also contains a variety of open-source libraries capable of tasks such as EKF (extended Kalman filtering) and computer vision and can provide an easy-to-configure interface for commonly-used sensors such as LiDAR and camera.

D3 Core Software Architecture

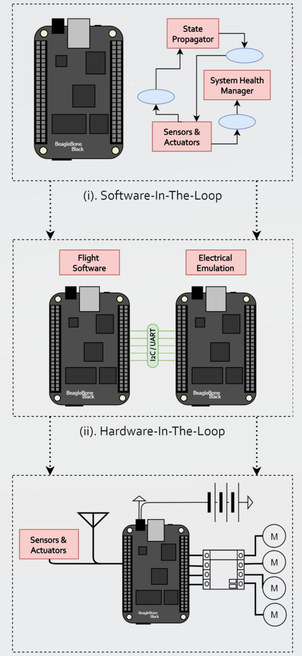


Conceptual ROS flowchart with nodes (red) and topics (blue) depicted.

Software Testing & Validation

The flight software testing and validation campaign consists of a three-layered approach:

i. **Software-in-the-loop (SWIL):** Core flight software nodes interact with simulated sensor/actuator and state propagation nodes on a single flight computer.
ii. **Hardware-in-the-loop (HWIL):** Core flight software nodes and simulated sensor nodes are operated on separate flight computers with a UART/I2C communication protocol emulation that mimics the desired hardware components.
iii. **FlatSat:** Core flight software nodes on the main flight computer interact with real hardware components distributed across an anti-static mat for probing of busses and powerlines.



For D3 SWIL and HWIL testing, on-orbit conditions are to be simulated in real-time for detumble (approx. 3 days) and guidance tracking (approx. 1-2 weeks) phases. Automated unit and integration testing ensures the individual nodes as well as node clusters are behaving nominally during this test campaign.

The basic structure for onboard D3 GNC simulations revolves around a state simulation node that interacts with the guidance tracker and magnetorquer actuator nodes. This state simulator uses Runge-Kutta (RK4) integration to propagate the vehicle state forward in real-time (1 Hz rate), with the same message/topic structure used for embedded hardware. These internal C++ simulations can be validated against their corresponding MATLAB equivalent, with small differences caused by different time-steps between each guidance state and differences in numerical integration, among other minor factors.