Learning to Code: How a Flight-Software Framework Helped a University Project Succeed

Bennett Research Technologies LLC

Michael Dompke (michael.dompke@slu.edu)

I. Introduction

A common problem among University-class missions is inadequate flight software development, leading to underperforming spacecraft, on orbit-failure, or even projects that never get off the ground. There are many possibilities for what is causing inadequate software; however, a primary cause is student training. For example, in the Space Systems Research Laboratory at Saint Louis University, undergraduate aerospace and mechanical engineers are primarily responsible for most software development. Majors like this have little to no practical software design training, and the programming they are taught is insufficient for flight software (e.g., MATLAB, LabView). For these reasons, we decided to adopt a flight-software framework containing collection of modules for developing platform control software (flight software)

II. Driving Factors for Adopting a Framework

Primarily Non-Computer Science Developers

- No prior experience with large Software Development

Beginner Friendly language (Python3)

- Many students do not have experience with low-level memory management (e.g., pointers, garbage collection). Python3 obscures this from the end-user.
- This interpreted language is easy to learn, allowing students to focus on software development and not learn about compilers.
- Develop an internal training program that teaches engineering students who know MATLAB about Python3 and how it compares to MATLAB.

Large Scale Software Development

- The framework is designed to include industry standard software verification and testing (Which was never apart of our previous missions)
- A framework allows our engineering team to focus on developing the spacecraft's software/hardware interfaces and control algorithms while not having to worry about the backend command and data handling.
- The compartmentalized design allows for engineers to work on a specific aspect of the flight software with having to worry about breaking all of it.

Space Systems Research Laboratory partnered with Bennett Research Technologies LLC, which developed the Artificial Reasoning for Exploration and Space (ARES) Framework. The ARES Framework has been used on one flight mission at Saint Louis University (Argus-02[SLU-03] Deployed Q1 2020) while currently being used on two missions in parallel development (DARLA[UNP NS-10 , CSLI] and DORRE[UNP NS-11]).

III. Benefits of the ARES Framework

The Artificial Reasoning for Exploration and Space (ARES) program is a framework for developing and operating intelligent networks of autonomous platforms such as spacecraft, robots, and fixed observation nodes. ARES includes inter-platform messaging and multi-process operations. ARES also include support for simple machine learning, image processing, rule-based processing, and other computing methods in understanding the environment of such intelligent networks.
ARES Framework is designed to run on a range of processors from large desktop or servers to small processors. In particular, ARES is designed to run on platforms and in environments with:

- Limited, low-bandwidth, intermittent communications
- Limited platform resources such as power
- Limited supervision, i.e. autonomy

ARES includes four key subsystems:

- ARES Core Subsystem (ACS) - provides the basic structure and communications services.
- ARES Mission Subsystem (AMS) - a near-complete example of a CubeSat platform including flight software. Can serve as a starting example for developing other autonomous platforms.
- ARES Management and Recovery Subsystem (AMR) - a set of services for managing multi-node software systems including processor/OS/software status, error detection, recovery and updates.
- Analytical Processing Subsystem (APS)- provides a framework for "reasoning" as stand-alone autonomous nodes or as a payloads in other autonomous platforms.

IV. Missions

Adopting the framework has allowed multiple missions to be developed in parallel. This allows for the development of core flight software, including the Master Control Program(MCP), Spacecraft Health Interface, Spacecraft Hardware Interface, Spacecraft ADCS, and Radio Communications Interface. Between each mission, these aspects of the core flight software are modified to fit the needs of the new mission. This allow for the DORRE and DARLA Flight Software to be developed in parallel.

Indexing a list and tuple

	MATLAB	MATLAB Example A = [2032,2074,1234]	Python	Python Example A = [2032,2074,1234]
Starts with	1	A(1) -> 2032	0	A[0] -> 2032
Last element of a list	end	A(end) -> 1234	-1	A[-1] -> 1234
n from end of list	end – (n-1)	A(end – (2 - 1)) -> 2074 Or A(end – 1) -> 2074	-n	A[-2] -> 2074

Example of MATLAB->Python Training Material

V. A Growing Team

SLU missions prior to the adaptation of the ARES Framework (COPPER/Argus-1) depended on 1-2 students with extensive prior extracurricular experience in embedded systems development. The projects depended on their willingness to put in long hours to build flight software from scratch, with no backup if they couldn't work. In the case of Argus-1, spacecraft development stalled for almost a year because the original team left and no replacements could be found

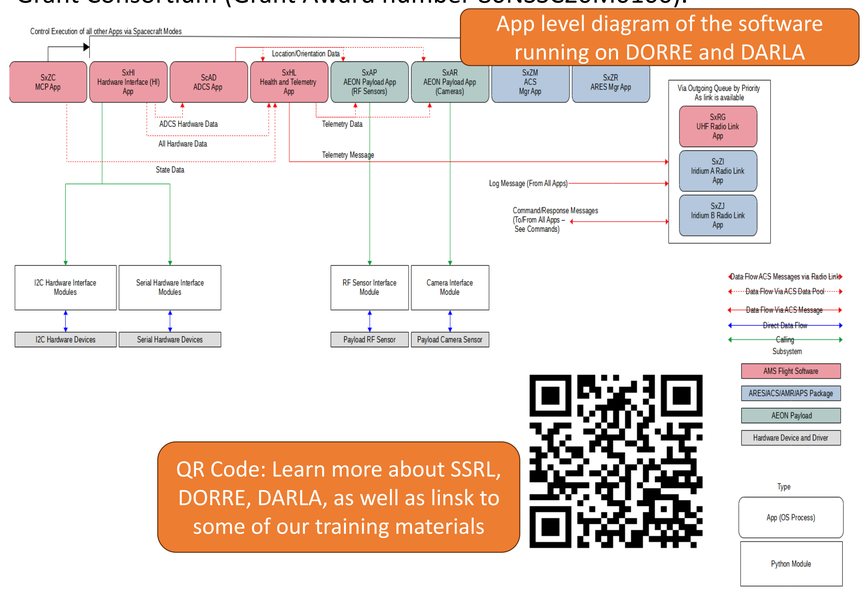
Python programing language has simplified our software onboarding process.

- All engineering students are required to take a class in MATLAB
- All computer Science (Major/Minor) Student are required to take a Python class
- Internal MATLAB to Python training material has been developed to teach Engineering Students Python
- Able to maintain a software team of ~6 members(Primarily Engineering and Computer Science)
- The app based architecture allows for small discrete task to be completed by engineering students in days or even hours, compared to weeks previously

VI. Acknowledgements

A massive thank you to Keith Bennett for creating the ARES Framework. Development of DARLA and DORRE was partially supported by the University Nanosat Program (NS-10 and NS-11 cycles, respectively) and the NASA Missouri Space Grant Consortium (Affiliate Awards, 2019-2022). This poster was supported in part by the NASA-Missouri Space Grant Consortium (Grant Award number 80NSSC20M0100).

App level diagram of the software running on DORRE and DARLA



QR Code: Learn more about SSRL, DORRE, DARLA, as well as linsk to some of our training materials