Development of a flight software framework for student CubeSat missions

Olman D. Quiros-Jimenez[1], Duncan d'Hemecourt[2]
[1] *Costa Rica Institute of Technology,* [2] *George Washington University*
[1] *olmanqj@gmail.com,* [2] *ddhemecourt@gwu.edu*

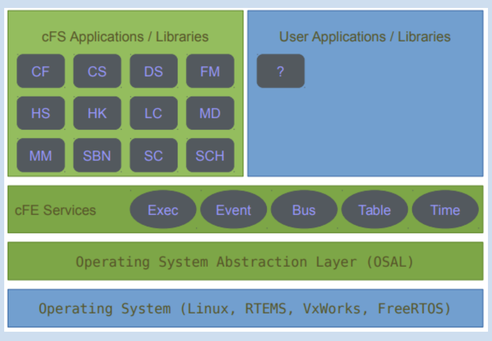
Introduction

- Student satellite missions, where students of different levels and fields carry out the design, integration, testing and operation phase of the spacecraft. This means that the mission is performed by non-experienced personnel, which can conduct to delays in the design and integration phase, and also to failures during the operation in space. The software component of satellites is not exempt from suffering this issues.
- There is not a rigorous way to design and implement the flight software. Although the existing heritage from past missions, sometimes the flight software is written from scratch [1].
- To address this, the paper attempts to give university students without previous experience, an initial reference when it comes to design and implement the flight software for CubeSats.

Available Open Source Flight Software Frameworks

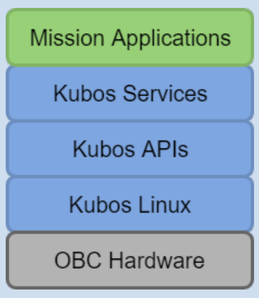
Some organizations have developed software frameworks to decrease the time and resources needed to implement the flight software, and improving the software quality.

cFS from NASA: is the more complete and mature, however this also brings a considerable learning curve. The memory footprint is not minimal, a typical configuration of cFS with FreeRTOS has a size around 1 MB [2], this can represent a limitation for low cost OBC modules.



[3]

KubOS from Kubos Corporation: allows to write the user application on Python or Rust. The KubOS platform runs above the KubOS-Linux distribution. This limits the portability in term of the OS, increases the memory footprint, and depending on the mission requirements, Linux can fail on providing real-time capabilities. Hardware portability is also limited, applications are portable only between the official supported COTS boards[4].



[4]

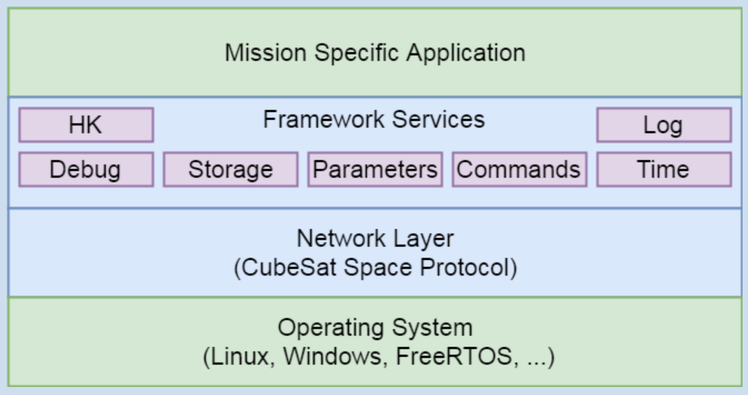
CubedOS from Vermont Tech: offers the capability for formal verification to be free of some types of run-time error. Can be executed without an OS, which reduces the memory footprint. However, is still a work in progress, and by the time this paper was written many features are still missing [5].

The three alternatives focus on modularity, reconfigurability and portability.

Flight Software Framework Proposal

We aim to propose an open-source software framework that can be easily understood and implemented by students in future academic CubeSat missions. Many of the current platforms available are either too complex for a simple embedded system, or provide too much of a learning curve for students.

Architecture: the proposed architecture consists of four layers: the operating system, the network layer (CubeSat Space Protocol), the framework services, and the user-specific software application.



Framework Services: based on the studied available frameworks, we can identify the fundamental modules that should be present in a flight software framework in order to be suitable for most CubeSat missions:

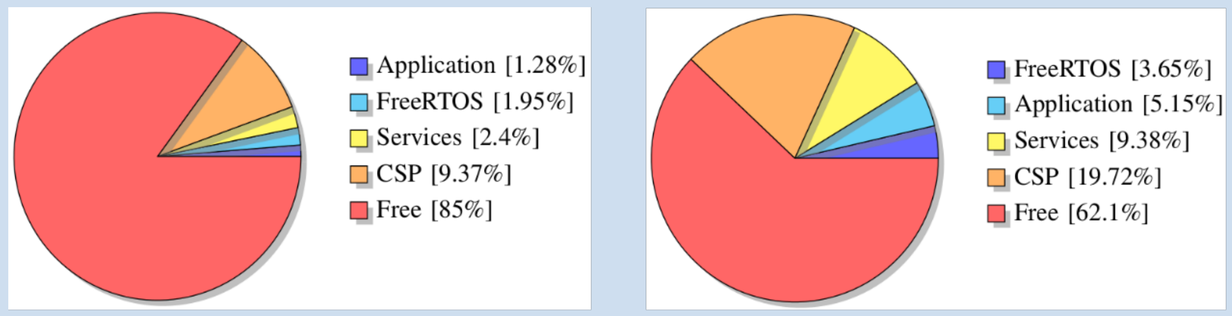
- Housekeeping Service: collect, store and broadcast telemetry data.
- Parameters Service: database for the spacecraft's parameters.
- Commands Service: handle commands from ground segment.
- Time Service: timekeeping related services.
- Storage Service: file system interface.
- Log Service: store events in log file.
- Debug Service: printing and reading to and from a debug console.

Implementation of Proposed Framework

The team demonstrated a functional flight software framework with the following attributes: very low memory footprint, modular, portable (OS and hardware) and real-time capabilities.

Platform: STK600 board and the 32-bits AT32UC3C0512C micro-controller, both by Microchip Technology Inc, with 512 KB ROM, 68 KB RAM, @ 16 MHz. For the operating system layer, FreeRTOS was selected by the team.

Memory Footprint: around 76 KB ROM and 26 KB RAM.



Memory allocation breakdown. ROM (left) total 15% (76.8 KB). RAM (right) total 37.9% (25.79 KB).

***Note**: the implementation is open-source, visit https://github.com/olmanqj/sfsf.*

References

[1] A. B. Ivanov and S. Bliudze, "Robust Software Development for University-Built Satellites," 2017.
[2] J. Wilmot, "Using CCSDS Standards to Reduce Mission Costs." [Online]. Available: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20170007440.pdf.
[3] A. Cudmore, G. Crum, S. Sheikh, and J. Marshall, "Big Software for SmallSats: Adapting cFS to CubeSat Missions,".
[4] Kubos Corporation, "Kubos Documentation," kubos.co, 2017. [Online]. Available: http://docs.kubos.co/1.2.0/index.html.
[5] "CubedOS Project Overview," cubesatlab.org. [Online]. Available: http://www.cubesatlab.org/CubedOS.jsp.