Work Done on the Operating Software for OpenOrbiter

Dayln Limesand
Timothy Whitney
Jeremy Straub
Ronald Marsh
The operating software team of the OpenOrbiter project has been tasked with developing software for general spacecraft maintenance, performing mission tasks and the monitoring of system critical aspects of the spacecraft. To do so, the team is developing an autonomous system that will be able to continuously check sensors for data, and schedule tasks that pertain to the current mission and general maintenance of the onboard systems. Development in support of these objectives is ongoing with work focusing on the completion of the development of a stable system. This poster presents an overview of current work on the project and highlights future objectives.

Overview

The OpenOrbiter Program aims to develop a template for a CubeSat spacecraft that can be used worldwide to reduce spacecraft development costs. Unlike other approaches, which may require $50,000 in upfront hardware costs or $250,000 in design expenses, an OPEN-class spacecraft can be built with a parts budget of under $5,000. This aims to enable low-cost educational missions and missions in developing regions.

To enable mission operations, a robust and lightweight task scheduler is required, due to CubeSats’ limited onboard power production capabilities as well as internal space restrictions. Schedules must be optimized; however, this optimization process must be performed within processing power constraints.

Several considerations must be taken into account in order to make a scheduler for these systems. This poster highlights requirements such as interdependency of onboard systems, limited windows of communication with ground based operators, and effects of current tasks on future tasks. After considering the genetic algorithm, exhaustive search, and heuristic based approaches, it has been decided that the heuristic based approach is the most feasible for this application. Time and power constraints were highly scrutinized in this comparison in order to maximize performance. Using this scheduling system will also enable the efficient collection of remote sensed data, such as images.

Work To-Date

With the limited power, processing, and time available on a small satellite’s flight computer (shown in Figure 1), it is vital to maximize the efficiency of the onboard scheduling software. On top of this, interdependencies of onboard systems, shown in Figure 2, were also addressed. Thorough qualitative evaluation of genetic, exhaustive search, and heuristic algorithms resulted in the selection of a heuristic approach. Based on this approach, an operating software flowchart (shown in Figure 3) has been created.

This approach takes into consideration the priority of tasks, how much power and time are used for each, if the task is in within the earliest start date and latest completion date and how many tasks will be affected if the current task is chosen. Once all of these have been taken into consideration, each task is assigned a weight. The system then begins running tasks in order of weight, beginning with the task with the lowest weight.

Operating Software Elements

The operating software is comprised of several elements which work in lock-step. These include:

- Scheduler - determines what will run at any given time; plans use of power and other resources of the spacecraft
- Hardware Connectors - communicate with sensors and actuators
- Radio / Communications - ground and payload

Acknowledgements

This paper is based on [8 and 9] which is revised and extended from Torgerson, et al’s “Scheduling for a Small Satellite for Remote Sensed Data Collection”

Future Work

Future work includes completion of both the heuristic scheduling system as well as the OpenOrbiter operating software, including communications, and systems monitor. Once complete, an assessment of the software based on NASA and Department of Defense Guidelines related to aerospace software systems and validation of compliance with Consultative Committee for Space Data Systems (CCSDS) communications standards (e.g., CCSDS 132.0-B-1 and CCSDS 133.0-B-1) will be performed. After initial launch we will begin to add greater support for new hardware and develop a community around an OPEN-class satellite.

References