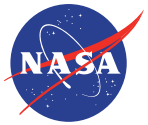


Jet Propulsion Laboratory
California Institute of Technology

Applying the F Prime Flight Software Framework to the ASTERIA CubeSat

Robert Bocchino
Brian Campuzano
Len Day

CubeSat Workshop
May 1, 2018
Cal Poly San Luis Obispo

ASTERIA CubeSat

Mission Overview

- Collaboration between JPL and MIT
- Space telescope in low-earth orbit
- Primary mission: Demonstrate technology
  - Precise and stable pointing of the imager
  - Stable thermal control of the focal plane
- Extended mission: Look for exoplanets

First for a CubeSat

CubeSats can do photometry

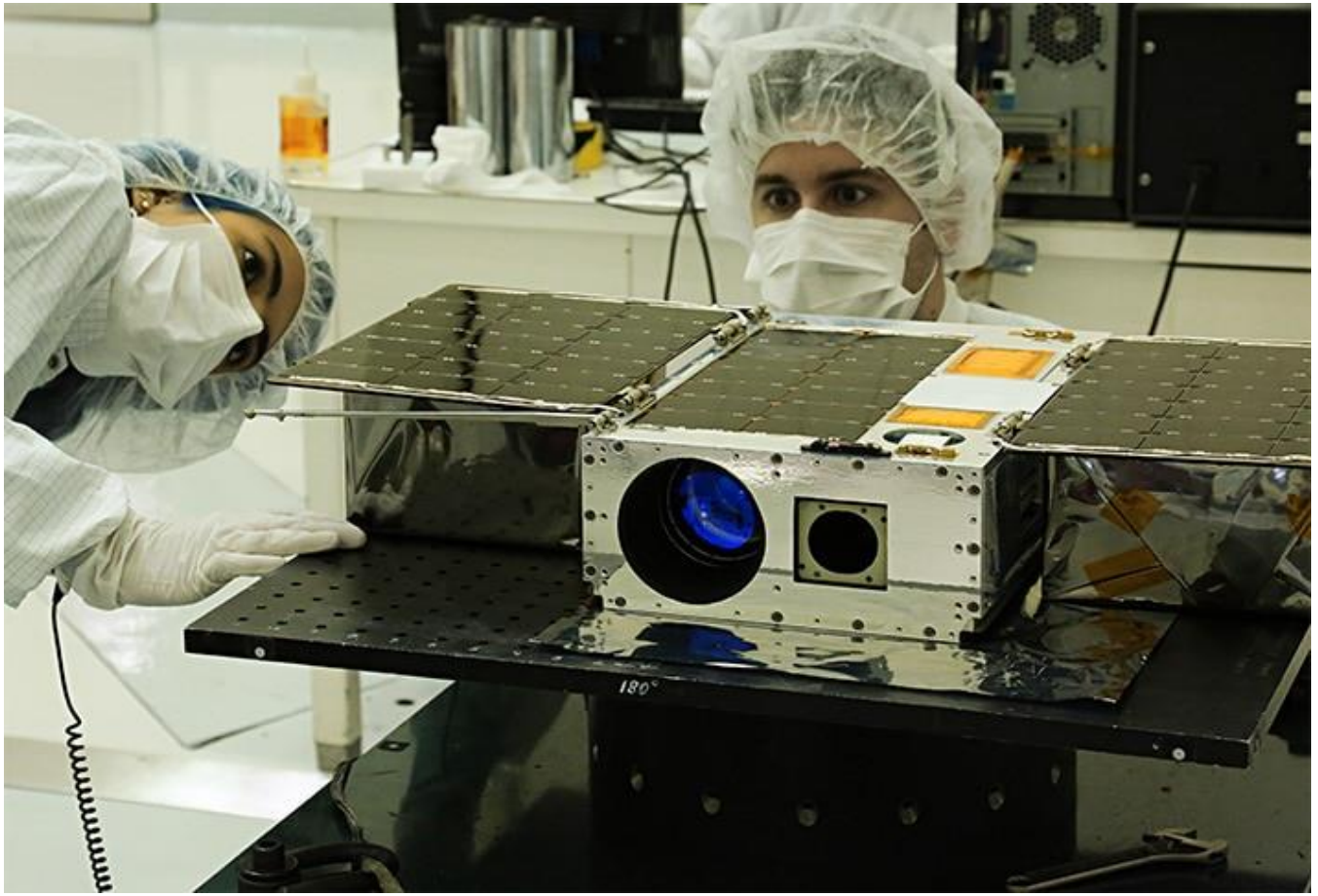
ASTERIA CubeSat

Spacecraft

- 6U CubeSat (~ 10 x 20 x 30 cm^3, 12 kg)
- Payload
  - Lens and baffle assembly
  - CMOS imager
  - Two-axis piezoelectric positioning stage
- Attitude and pointing control
  - Coarse grain: BCT XACT (reaction wheels)
  - Fine grain: Closed-loop control of piezo stage
- Thermal control
  - Passive cooling
  - Heaters driven by closed-loop control

ASTERIA CubeSat

Spacecraft



https://www.jpl.nasa.gov/cubesat/missions/asteria.php

ASTERIA CubeSat

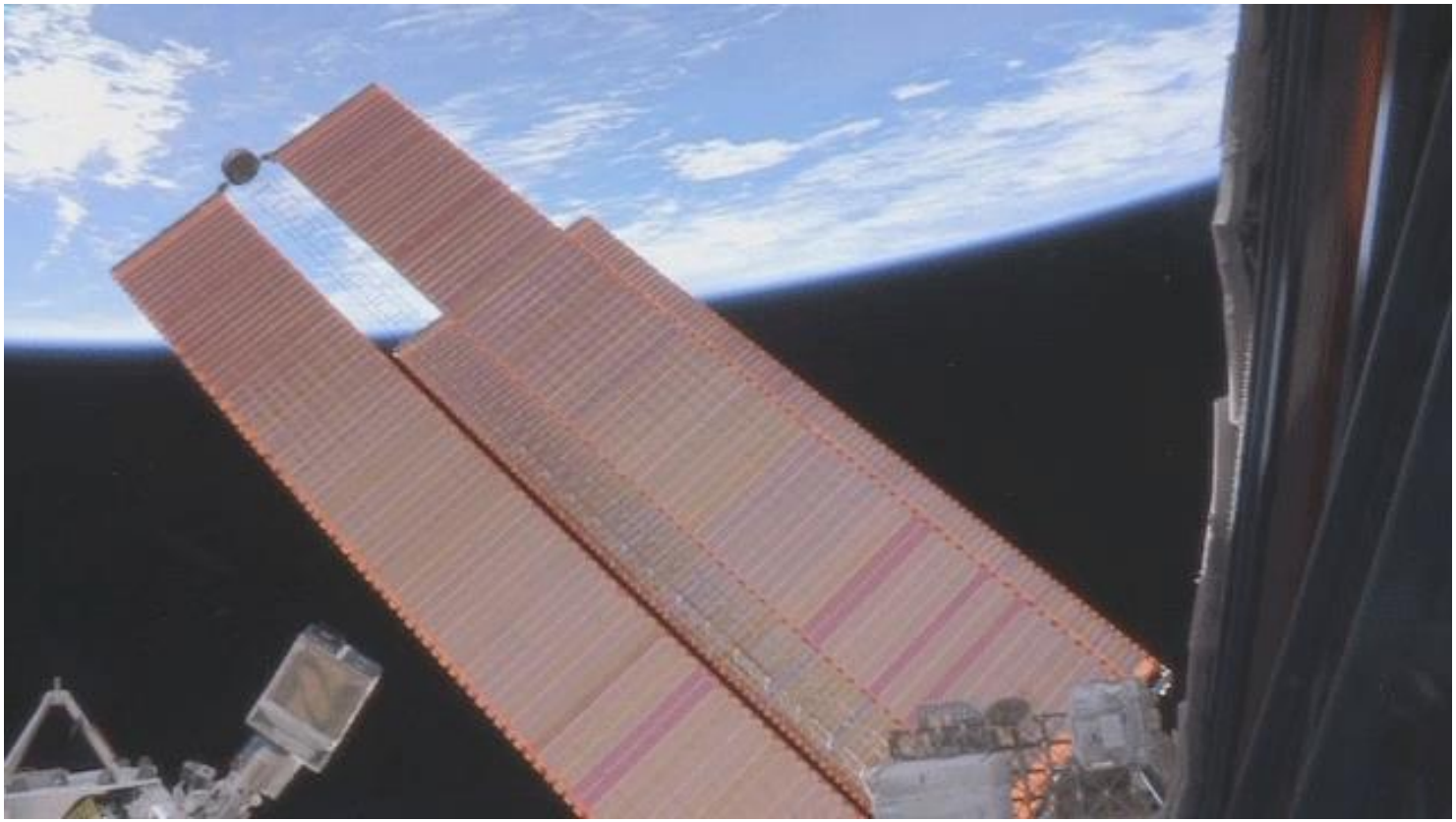
Mission Timeline

Time	Event
June 2017	Delivery to NanoRacks CubeSat deployer
August 2017	Launch and delivery to International Space Station (ISS)
November 2017	Deployment from ISS
February 2018	Completion of 90-day primary mission
Through May 2018	In extended mission

All mission goals achieved

ASTERIA CubeSat

Deployment



https://www.jpl.nasa.gov/cubesat/missions/asteria.php

ASTERIA Flight Software (FSW)

Overview

- Ten subsystems
- 201K lines of code
  - 56% auto-generated
  - 25% inherited
- ~6 person-years of effort over 2.4 years
- Written mostly in C++
  - Drivers written in C
  - 3 of 12 drivers provided by hardware vendors
- Uses the F Prime FSW framework

F Prime Flight Software Framework

Overview

- Free and open-source; developed at JPL
- Tailored to small-scale systems
  - CubeSats, SmallSats, instruments
- Comprises several elements
  - A component-based architectural approach
  - A C++ framework providing core capabilities (e.g., queues)
  - Tools for specifying models and generating code
  - A collection of ready-to-use components
  - Tools for unit and integration testing
- Supports a wide range of hardware platforms
- Runs on several OSs (e.g., Linux, Mac OS, VxWorks)

https://github.com/nasa/fprime

F Prime Flight Software Framework

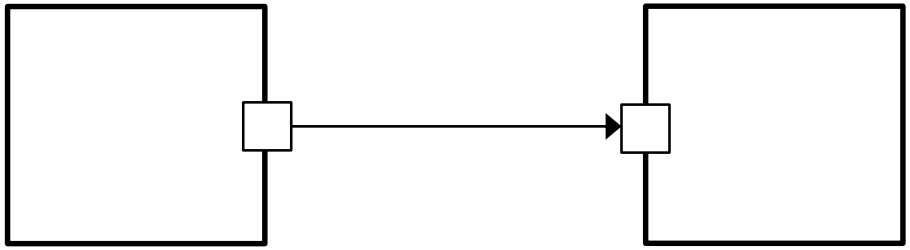
Architecture

- Based on components and ports
  - **Component:** A unit of FSW function (like a C++ class)
  - **Port:** A connection to other component instances
- Component instances
  - May be active (internally threaded) or passive (no thread)
  - Communicate only through ports
  - Have no compile-time dependencies on other components
- Port connections
  - Are typed and statically specified
  - May be synchronous or asynchronous

Provides structure to FSW applications
Enables automatic checking of correctness properties
Enhances reusability of FSW components

F Prime Flight Software Framework

Modeling and Code Generation

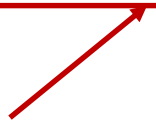


- FSW engineers write a high-level model
  - Define components and ports
  - Specify connections in **topology** (connection graph)
  - Define commands, telemetry, and events
- Tools generate
  - Generic part of component code
    - E.g., Code for receiving and decoding commands
    - FSW engineers fill in handlers with C++ code
  - Code for connecting the ports
  - Command, telemetry, and event dictionaries

ASTERIA Flight Software (FSW)

Components and Deployments

- 54 components
  - 17 inherited from F Prime
  - 22 developed for ASTERIA and reusable in future missions
  - 16 developed for ASTERIA and mission-specific
- 93 component instances
- 17 deployments (executable component configurations)
  - 12 subsystem deployments
  - 5 release deployments



Adaptable to different hardware configurations in test

ASTERIA Flight Software (FSW)

Reusable and Mission-Specific Components

Subsystem	Reusable Components	Mission-Specific Components
Attitude Control (includes coarse pointing)	No	Yes
Communication	Yes	Yes
Engineering	Yes	Yes
Fault Protection	Yes	Yes
Health Monitoring	Yes	No
Mode Management	No	Yes
Pointing Control (fine pointing)	No	Yes
Power Management	No	Yes
Solar Array Deployment	No	Yes
Thermal Control	Yes	Yes

ASTERIA Flight Software (FSW)

New Toolchain Capabilities

- We added the following to F Prime for ASTERIA
  - Support for automated integration testing in python
  - Lightweight modeling based on text input
  - Subsystem topologies
    - Can run on their own
    - Can be imported into release topologies
  - Enhanced auto-coding
    - Structure, array, and enumeration types
    - Health monitoring connections
    - Setup code for multiple deployments
    - Code for sending commands between components
- Automated integration testing is part of mainline F Prime
- We are working to integrate the other features

ASTERIA Flight Software (FSW)

Lessons Learned

- F Prime helped us get this job done
  - Architectural structure and patterns
  - Modeling and code generation
  - Direct component reuse
- System design
  - Layered architecture worked well
  - It would have helped to add a test mode to the mode manager
  - Invoking output ports while holding a lock can cause deadlock
    - Suggests need for concurrency analysis in F Prime
    - This is possible because of the static connection graph
- Component design
  - Provide explicit commands for connect/disconnect to hardware
  - Components should not re-initialize themselves in fault responses

Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov