Jet Propulsion Laboratory
California Institute of Technology

F Prime Flight Software Product Line Overview

Jeff Levison, Supervisor
Small Scale Flight Software Group
Jet Propulsion Laboratory

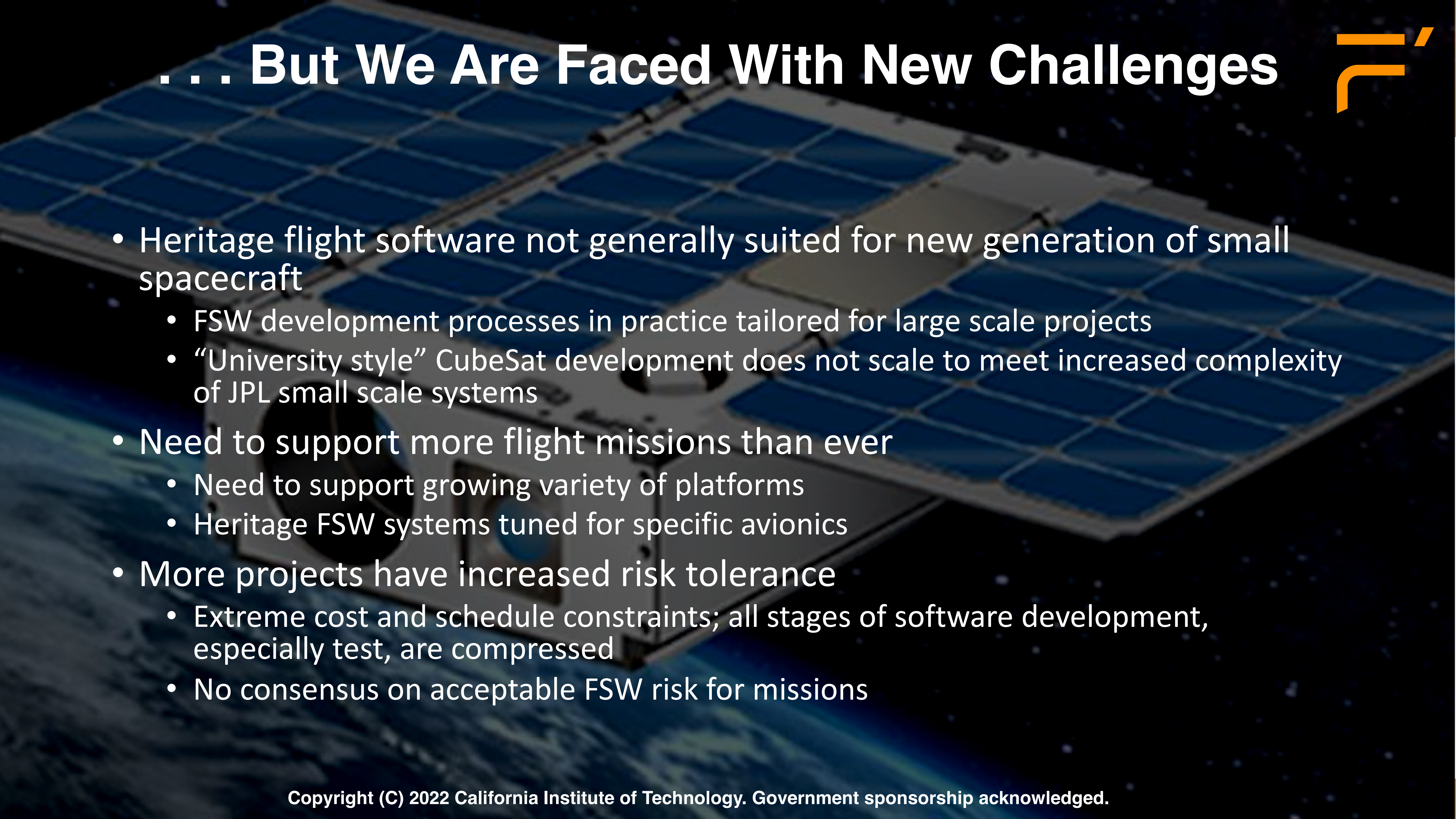
May 2, 2022

JPL Has a Successful Legacy Addressing Flight Software Challenges . . .

- The onboard software operates the entire spacecraft – across all its subsystems – continuously throughout its mission
- It has to work! Long one way light times and intermittent communications require reliable, and capable software that is able to safe the spacecraft and phone home when there are failures
- Mission critical events like a comet flyby or Mars entry, descent and landing require software autonomy sufficient to carry out the activity successfully without ground-in-the-loop intervention

- Long missions require on-board spacecraft capability to reload/update flight software in flight
- Intermittent contacts require on-board data storage and management
- Increasingly complex missions and associated flight hardware have resulted in more capability being included in software
- Significant effort allocated to software verification and validation conducted by a V&V team independent of the development team
- Modest on-board autonomy requires large operations teams

. . . But We Are Faced With New Challenges



- Heritage flight software not generally suited for new generation of small spacecraft
  - FSW development processes in practice tailored for large scale projects
  - “University style” CubeSat development does not scale to meet increased complexity of JPL small scale systems
- Need to support more flight missions than ever
  - Need to support growing variety of platforms
  - Heritage FSW systems tuned for specific avionics
- More projects have increased risk tolerance
  - Extreme cost and schedule constraints; all stages of software development, especially test, are compressed
  - No consensus on acceptable FSW risk for missions

The F Prime (F′) Flight Software Product Line

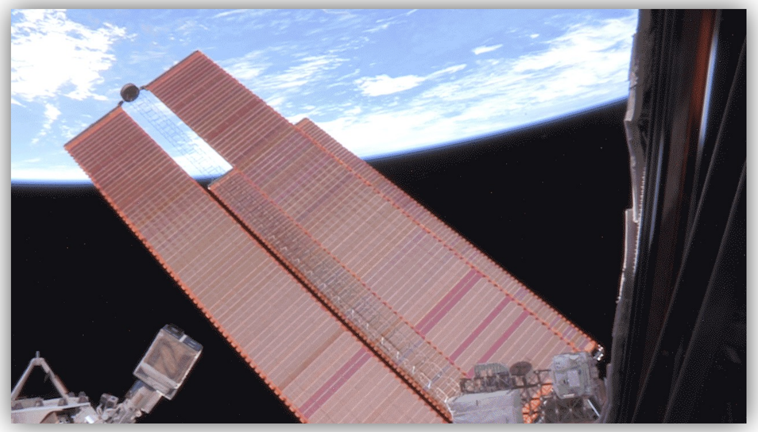
- The F Prime Product Line targets small scale, in-house missions that are expected to fall in three general categories:
  - Small-scale flight systems, i.e. CubeSats or SmallSats
  - Flight instruments and/or sensors requiring software developed for a dedicated processing platform
  - Flight "deployables" – freely operating spaceborne flight systems that are deployed by spacecraft in support of a mission (ex. a planetary lander or helicopter)
- Additionally, F′ supports
  - Testbed/prototyping efforts requiring embedded software
  - Software technology development with intended path to flight
  - Collaborative flight software development with universities and industry partners and among JPL small scale projects in concurrent development
- The F′ Product Line can support a wide variety of projects and tasks

What is F′?

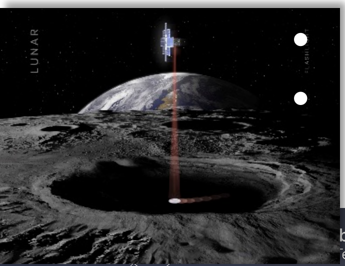
- F′ facilitates the development of software for embedded applications
- Released 2017 as open-source software on NASA GitHub: https://github.com/nasa/fprime
- The Product Line consists of several elements:
  - An architectural approach that decomposes an embedded software system into discrete components with well-defined interfaces that communicate over ports interconnected in a topology
  - A C++ framework that provides basic features such as message queues, threading and an OS abstraction layer
  - A suite of tools for specifying components and their connections and automatically generating a partial implementation from the specification
  - A growing collection of generic components for basic features such as command dispatch, event logging, and memory management that can be incorporated without modification into new software projects
  - A suite of tools for testing software at the unit and integration levels including a lightweight Ground Data System
- F′ is managed independently from flight projects ensuring all elements are maintained regularly using best practices

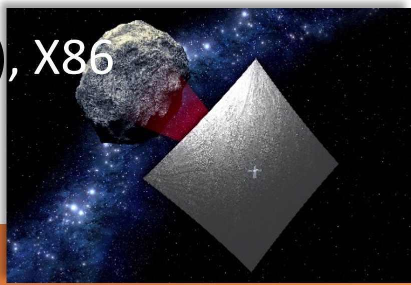
Use At JPL Is Increasing

- Developed under a JPL technology exploration task (2013)
- Previously flown
  - RapidScat – radar scattermometer on ISS (2014-2016)
  - ASTERIA CubeSat mini space telescope (2017-2019)
- Currently in operations on Mars Helicopter (Deployed April 3, 2021)
- Baselined flight software deployments in development
  - Lunar Flashlight CubeSat
  - Near Earth Asteroid Scout CubeSat
  - Ocean Worlds Life Surveyor instrument
  - CADRE mini lunar rover demo
  - COLDArm deployable lunar arm
  - Mars Sample Return Lander Motor Controller



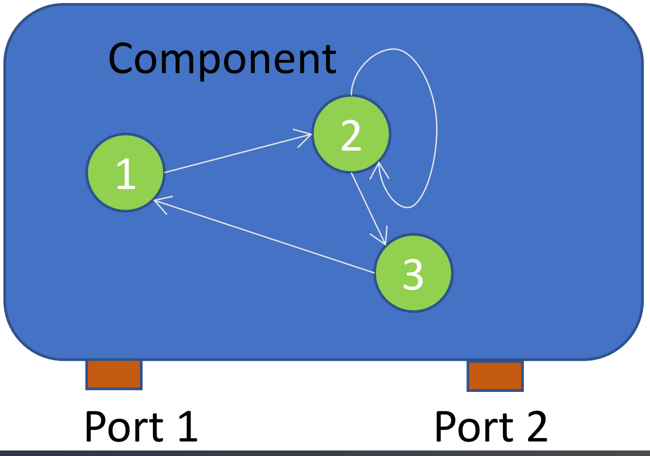
- Used or planned for several technology tasks in Divs 31, 33 and 34
- Demonstrated on a variety of platforms
  - Processor architectures: Power PC (Rad750), ARM, TI MSP430, Leon 3 and 4(SPARC), X86
  - Operating systems: Linux, VxWorks, RTEMS, MacOS, Raspberry Pi OS, bare metal



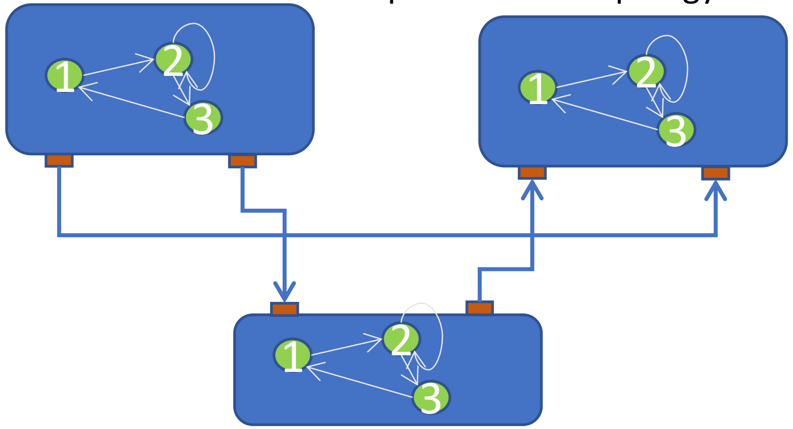


F′ Provides a Powerful Software Architecture

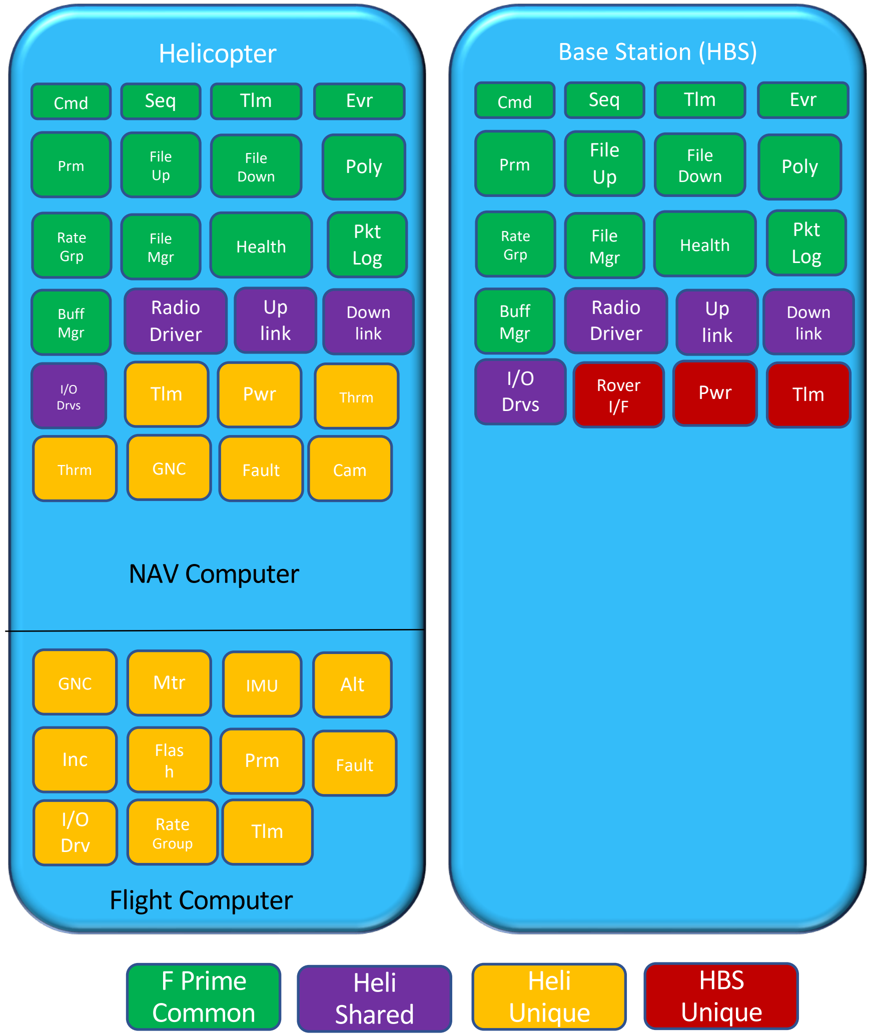
- F Prime is a component architecture
- Software is composed of ***components*** (behaviors) and ***ports*** (interfaces between components)
- Components are interconnect together to form ***topologies***, which comprise the binary built as a ***deployment***.
- Components are not link dependent on other components, so they can be easily recombined to form alternate topologies.
- Architectural features: rapid development, portability, high performance, component reusability, software system analyzability and testability
- Mars Helicopter had 11 different deployments for various test venues and ground support applications.



Connected components in a topology



Mars Helicopter Flight Software Components



Helicopter

NAV Computer

Cmd, Seq, Tlm, Evr
Prm, File Up, File Down, Poly
Rate Grp, File Mgr, Health, Pkt Log
Buff Mgr, Radio Driver, Up link, Down link
I/O Drvs, Tlm, Pwr, Thrm
Thrm, GNC, Fault, Cam

Flight Computer

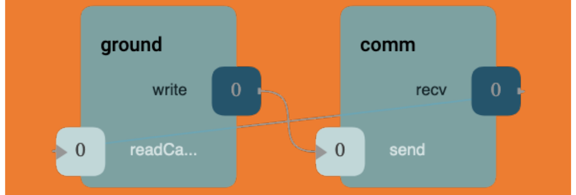
GNC, Mtr, IMU, Alt
Inc, Flas h, Prm, Fault
I/O Drv, Rate Group, Tlm

Base Station (HBS)

Cmd, Seq, Tlm, Evr
Prm, File Up, File Down, Poly
Rate Grp, File Mgr, Health, Pkt Log
Buff Mgr, Radio Driver, Up link, Down link
I/O Drvs, Rover I/F, Pwr, Tlm

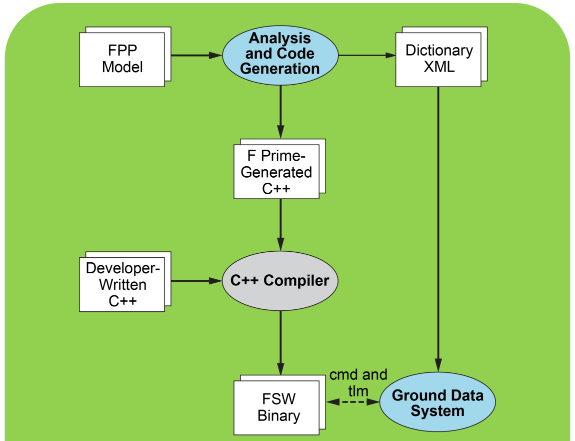
🟩 F Prime Common
🟪 Heli Shared
🟨 Heli Unique
🟥 HBS Unique

F′ Streamlines Phases of Software Development



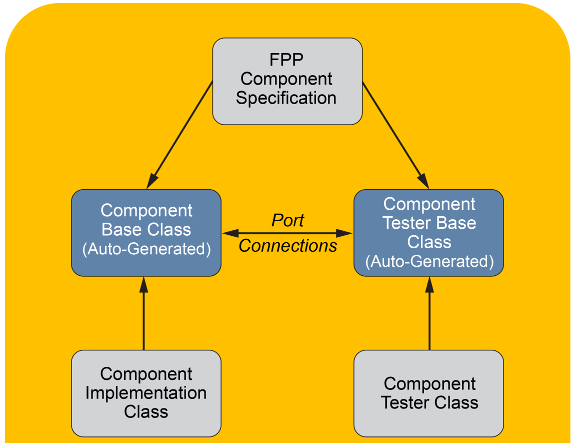
Design

- A domain-specific modeling language, F Prime Prime (FPP) provides modeling front end for specifying components, ports and topologies
- FPP can also specify component commands, telemetry, events and paramters for a ground dictionary
- Companion tools provided for visualizing topology graphs
- Analysis tools check correctness of models prior to code generation



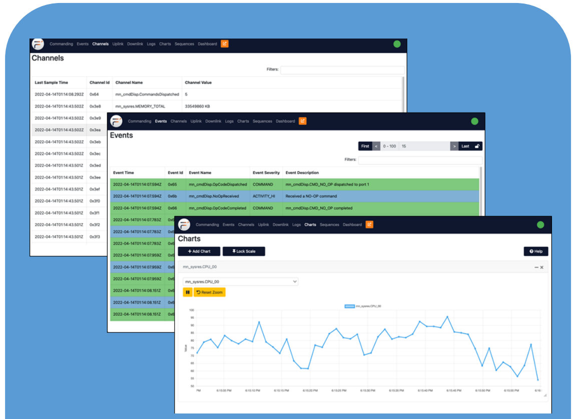
Implementation

- Autocode tools translate FPP models in to C++, representing the bulk of the structural code for the software system
- Users develop the behavioral code for the components


FPP Component Specification
Component Base Class (Auto-Generated)
Port Connections
Component Tester Base Class (Auto-Generated)
Component Implementation Class
Component Tester Class

Unit Testing

- Auto-generated test classes provide mirrored interfaces to modeled components for developing unit tests
- Rules-based testing option further streamlines test construction and improves reusability

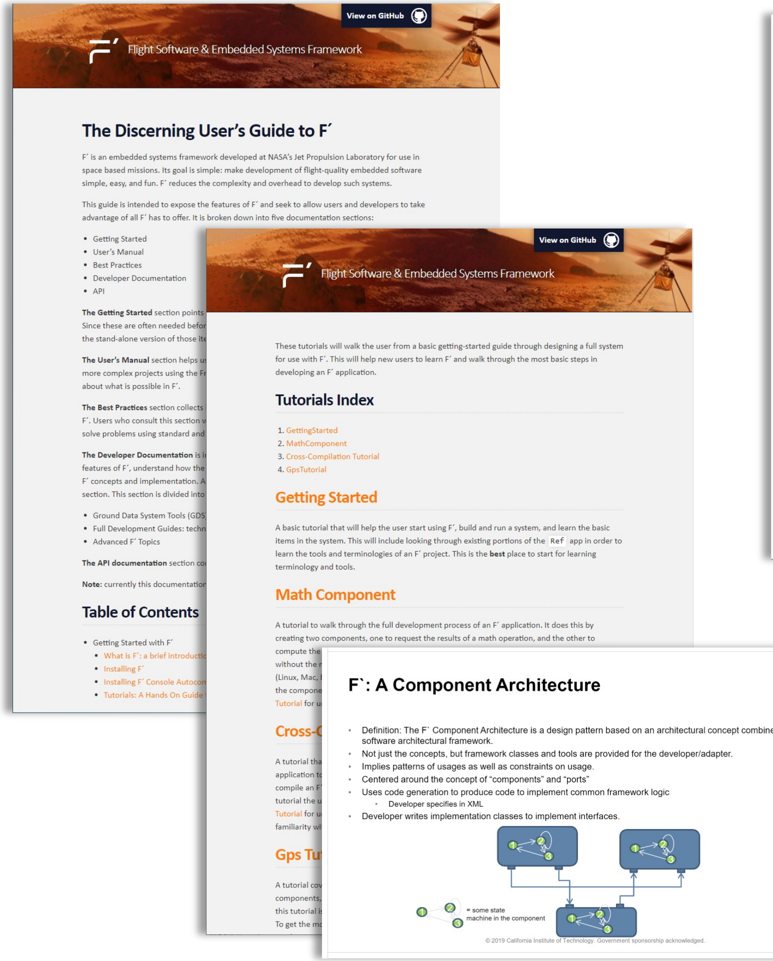


Integrated Testing

- Integrated Python-based Ground Data System included supporting integration and system level testing
  - Incorporates auto-generated command and telemetry dictionary
  - Integration and test API provided for writing automated tests
  - Web-based GUI for sending commands, viewing telemetry and events and simple plots
- F Prime can be adapted to support other GDSs employed on a project

F′ Includes User Documentation and Support

Online User Guides, Tutorials, Specs and FSW Workshop Materials


View on GitHub
Flight Software & Embedded Systems Framework
View on GitHub
Flight Software & Embedded Systems Framework

The Discerning User's Guide to F′

F′ is an embedded systems framework developed at NASA's Jet Propulsion Laboratory for use in space based missions. Its goal is simple: make development of flight-quality embedded software simple, easy, and fun. F′ reduces the complexity and overhead to develop such systems.

This guide is intended to expose the features of F′ and seek to allow users and developers to take advantage of all F′ has to offer. It is broken down into five documentation sections:

- Getting Started
- User's Manual
- Best Practices
- Developer Documentation
- API

The Getting Started section points
Since these are often needed befor
the stand-alone version of those ite

The User's Manual section helps us
more complex projects using the Fr
about what is possible in F′.

The Best Practices section collects
F′. Users who consult this section w
solve problems using standard and

The Developer Documentation is i
features of F′, understand how the
F′ concepts and implementation. A
section. This section is divided into

- Ground Data System Tools (GDS
- Full Development Guides: techn
- Advanced F′ Topics

The API documentation section co

Note: currently this documentation

Table of Contents

- Getting Started with F′
  - What is F′: a brief introductio
  - Installing F′
  - Installing F′ Console Autocom
  - Tutorials: A Hands On Guide

These tutorials will walk the user from a basic getting-started guide through designing a full system for use with F′. This will help new users to learn F′ and walk through the most basic steps in developing an F′ application.

Tutorials Index

1. GettingStarted
2. MathComponent
3. Cross-Compilation Tutorial
4. GpsTutorial

Getting Started

A basic tutorial that will help the user start using F′, build and run a system, and learn the basic items in the system. This will include looking through existing portions of the `Ref` app in order to learn the tools and terminologies of an F′ project. This is the **best** place to start for learning terminology and tools.

Math Component

A tutorial to walk through the full development process of an F′ application. It does this by creating two components, one to request the results of a math operation, and the other to compute the
without the r
(Linux, Mac, E
the compone
Tutorial for u

Cross-C

A tutorial tha
application to
compile an F′
tutorial the u
Tutorial for u
familiarity wi

Gps Tu

A tutorial cov
components,
this tutorial is
To get the mo

F`: A Component Architecture

- Definition: The F` Component Architecture is a design pattern based on an architectural concept combined with a software architectural framework.
- Not just the concepts, but framework classes and tools are provided for the developer/adapter.
- Implies patterns of usages as well as constraints on usage.
- Centered around the concept of "components" and "ports"
- Uses code generation to produce code to implement common framework logic
  - Developer specifies in XML
- Developer writes implementation classes to implement interfaces.

© 2019 California Institute of Technology. Government sponsorship acknowledged.

Publications

SSC-18-XII-04

F Prime: An Open-Source Framework for Small-Scale Flight Software Systems

Robert L. Bocchino Jr., Timothy K. Canham, Garth J. Watney, Leonard J. Reder, Jeffrey W. Levison
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, Pasadena, CA 91109–8099; (818) 354–8175
Robert.L.Bocchino@jpl.nasa.gov

ABSTRACT

Developing flight software for small-scale missions such as CubeSats and SmallSats is challenging. These missions typically have ambitious goals, modest budgets, and tight schedules. To meet these challenges, a good flight software framework is essential. Frameworks can provide an architecture, infrastructure, tools, and reusable software components, all of which can help developers deliver their code on time and on budget.

In this paper we present F Prime, a free, open-source flight software framework developed at JPL and tailored to small-scale systems such as CubeSats, SmallSats, and instruments. F Prime comprises several elements: (1) an architecture that decomposes flight software into discrete components with well-defined interfaces; (2) a C++ framework that provides core capabilities such as message queues and threads; (3) tools for specifying components and connections and automatically generating code; (4) a growing collection of ready-to-use components; and (5) tools for testing flight software at the unit

We describe the F Prime fra
ments to the framework curr
systems for testing.

1. INTRODUCTION

Developing spacecraft fligh
lenging under any circumsta
lenging for a small, cost-con
tious goals. Many small sa
SmallSats) fit into this cate
stages of the development c
be compressed. FSW deve
such as insufficient staff, ina
hardware, poorly specified
fied and rapidly changing
problems exist to some degr
are more pronounced in t
strained, rapid deployment.

FPP: A Modeling Language for F Prime

Robert L. Bocchino Jr., Jeffrey W. Levison, and Michael D. Starch
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
{robert.l.bocchino,jeffrey.w.levison,michael.d.starch}@jpl.nasa.gov

Abstract—**We present F Prime Prime (FPP), a new open-source modeling language for F Prime. F Prime is an open-source flight software framework developed at JPL and deployed, among other places, on the Mars helicopter *Ingenuity*. FPP provides a convenient way to model the architectural elements of an F Prime application, e.g., components, ports, and their connections. It has a succinct and readable syntax, a well-defined semantics, and robust error checking and reporting. The FPP tool suite, written in Scala, analyzes FPP models, reports errors, and translates correct FPP models to a combination of XML and C++. Existing F Prime tools translate the XML to a partial implementation in C++, to be completed by the developers. The model elements have clean interfaces and are highly reusable. An accompanying visualization tool constructs diagrams of components and connections that FSW developers can use to understand and communicate their designs, for example at reviews. We discuss the design and implementation of FPP and the integration of FPP into F Prime. We also discuss our experience using FPP to construct F Prime models. Finally, we discuss our plans for future work, including improved code generation, improved visualization, and more advanced analysis capabilities.**

TABLE OF CONTENTS

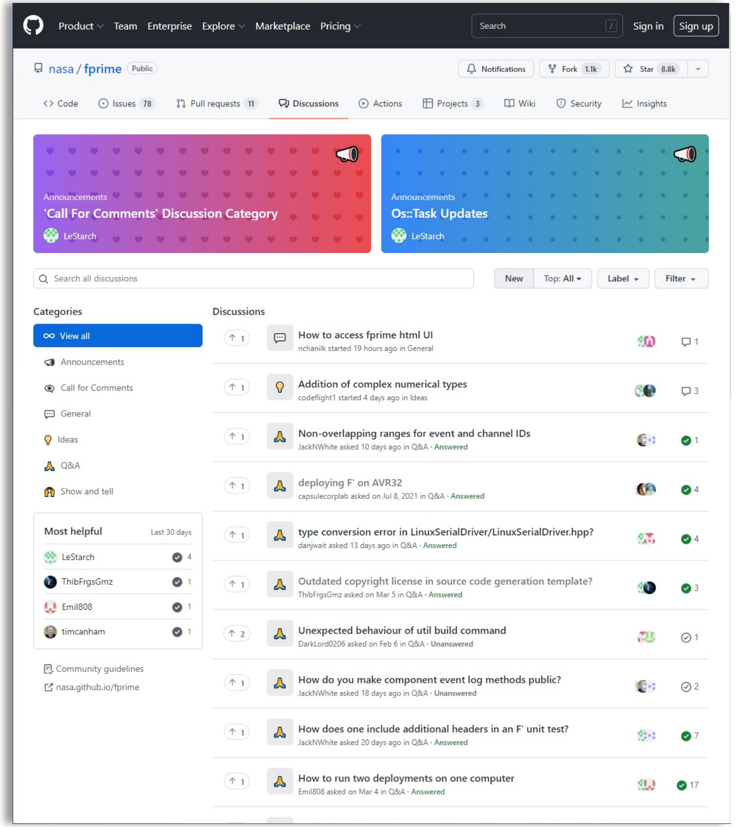
cessfully deployed on several missions. One such mission is the Mars helicopter *Ingenuity*, the first-ever powered aircraft to fly on another planet [2][28]. Other recent and upcoming missions include the ASTERIA space telescope [31][29], the Lunar Flashlight mission to the Moon [3], and the NEA Scout mission to a near-earth asteroid [4].

F Prime uses a **model-based** approach to FSW development. In this approach, developers describe the high-level structure of the software in a **model**, which is a machine-representable data structure with a formal specification. Using a model has several advantages: it provides a clear statement of design intent, it can be visualized as a set of diagrams, it can be automatically checked for correctness properties, and it can form the input to tools that auto-generate a partial FSW implementation in C++.

This approach requires a **modeling tool**, i.e., a way for developers to express the model. The modeling tool can be a domain-specific language, a graphical tool, or a combination of the two. Before F Prime v3.0.0 (released December 22, 2021), the preferred modeling tool for F Prime was Extensible Markup Language (XML) extended with an F Prime-specific schema and a code generator written in Python. Using XML has some advantages: it is widely used, lightweight, easy to parse, and flexible. However, the F Prime XML schema has some major disadvantages as a modeling language: it is verbose and inconvenient for users to read and write, its semantics is not well specified, and it has poor error checking and reporting. Many errors go through to be caught by the C++ compiler. There are also places in the model where code could be generated but is not; these gaps have to be filled in with handwritten C++.

In this paper we present F Prime Prime (FPP) [5]. FPP is a new open-source domain-specific language for constructing,

Supported User Discussion Forum and Issue Tracking



Announcements
'Call For Comments' Discussion Category
LeStarch

Announcements
Os::Task Updates
LeStarch

Categories

- Announcements
- Call for Comments
- General
- Ideas
- Q&A
- Show and tell

Most helpful	Last 30 days
LeStarch	✓ 4
ThibFrgsGmz	✓ 1
Emil808	✓ 1
timcanham	✓ 1

Community guidelines
nasa.github.io/fprime

Discussions

Votes	Discussion	Replies
↑ 1	How to access fprime html UI nchanilk started 19 hours ago in General	1
↑ 1	Addition of complex numerical types codeflight1 started 4 days ago in Ideas	3
↑ 1	Non-overlapping ranges for event and channel IDs JackNWhite asked 10 days ago in Q&A · Answered	✓ 1
↑ 1	deploying F' on AVR32 capsulecorplab asked on Jul 8, 2021 in Q&A · Answered	✓ 4
↑ 1	type conversion error in LinuxSerialDriver/LinuxSerialDriver.hpp? danjwait asked 13 days ago in Q&A · Answered	✓ 4
↑ 1	Outdated copyright license in source code generation template? ThibFrgsGmz asked on Mar 5 in Q&A · Answered	✓ 3
↑ 2	Unexpected behaviour of util build command DarkLord0206 asked on Feb 6 in Q&A · Unanswered	1
↑ 1	How do you make component event log methods public? JackNWhite asked 18 days ago in Q&A · Unanswered	2
↑ 1	How does one include additional headers in an F' unit test? JackNWhite asked 20 days ago in Q&A · Answered	✓ 7
↑ 1	How to run two deployments on one computer Emil808 asked on Mar 4 in Q&A · Answered	✓ 17

Enabling Software Technology Advancement

- **Software Design:** worked with Carnegie Mellon to replace prior modeling and auto-code tools requiring SysML/MagicDraw plugin with revisions based on F Prime Prime (FPP), a modeling specification tailored to F', delivered as part of version 3.0.0 open source release in December 2021
- **Software Analysis:** working with CMU to extend FPP to model performance
- **Software Verification:** expanding scenario-based testing to automatically construct and execute tests, resulting in a structured approach to testing that generates many more tests than could explicitly be constructed by hand
- **Next Gen Avionics Development:** F' baselined for Sphinx/Coyote and Sabertooth in-house processing platforms for CubeSats and SmallSats, respectively
- **Mission enhancement:** F' supported a 3 year autonomy R&TD project integrating autonomous planning and execution into all levels of flight software that led to an opportunistic flight demo on ASTERIA

F' enables rapid advancement of state of the art FSW and mission capability with direct infusion path to projects

Enabling Educational Outreach

- Conducted Flight Software/F’ workshops and presentations for students on several campuses
- Three F’ CMU masters student projects funded by SURP; UM SURP proposed; Bowie task funded by ESD
- Over 25 student internships funded
- Supporting CMU, Georgia Tech, Cal Poly Pomona & SLO, UNLV and U. Alabama with their student CubeSat or similar programs using F’
- Georgia Tech GT-1 CubeSat with F’ successfully deployed from ISS 2/3/22

F’ fits into a NASA recruiting strategy specifically targeting computer science, computer engineering and software engineering undergraduate and graduate students

Carnegie Mellon University

UNLV

AUBURN UNIVERSITY

UNIVERSITY OF MICHIGAN

BOWIE STATE UNIVERSITY 1865

CalPoly Pomona

CORNELL UNIVERSITY FOUNDED A.D. 1865

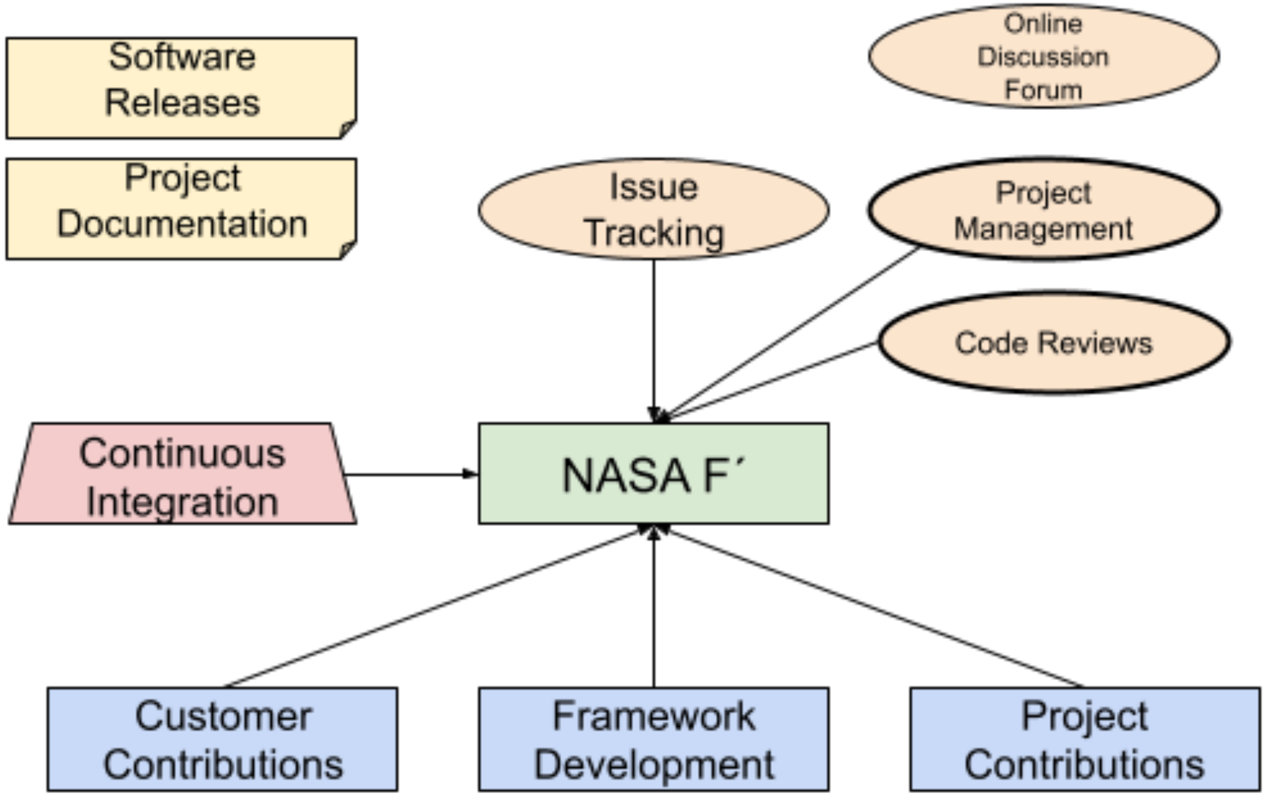
UAH THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

CU Boulder

CAL POLY SAN LUIS OBISPO

Embracing Openness for a Growing User Community

- All core product line functions have been moved external, into the open
  - Reduces maintenance overhead
  - Fosters a stronger community with better collaboration and oversight
- Active user engagement - over 1,000 forks and over 100 contributors in GitHub
- Visibility from Mars Helicopter and release of F′ V3.0.0 has sparked more inquiries within aerospace industry



Summary

- F Prime is a managed, flight-proven software product line developed to strategically meet the delivery challenges associated with a growing number of flight/embedded tasks
- F Prime is designed for efficient, collaborative flight software development resulting in successful projects utilizing much more than a defined software architecture and reusable flight code
- We have demonstrated ability to advance state of the art flight software technology development and infuse results into projects
- Accessible open source software engages a growing set of internal and external users supporting embedded projects

Flight Software Development Workshop

October 17 – 19, 2022

Jet Propulsion Laboratory, Pasadena, CA

Free - Invitation Only, Limited Availability

Topics Covered

- Flight Software System Engineering
- Flight Software Architecture Principles
- Flight Software Design
- Coding Style
- Basic Data Structures
- Unit Testing
- Integrated Testing with a Ground Data System
- Lab assignments featuring the F Prime Flight Software Product Line

If interested send email to fprime@jpl.nasa.gov

Thank You