OpenSatKit

A Tool Suite for Working with NASA's Core Flight System

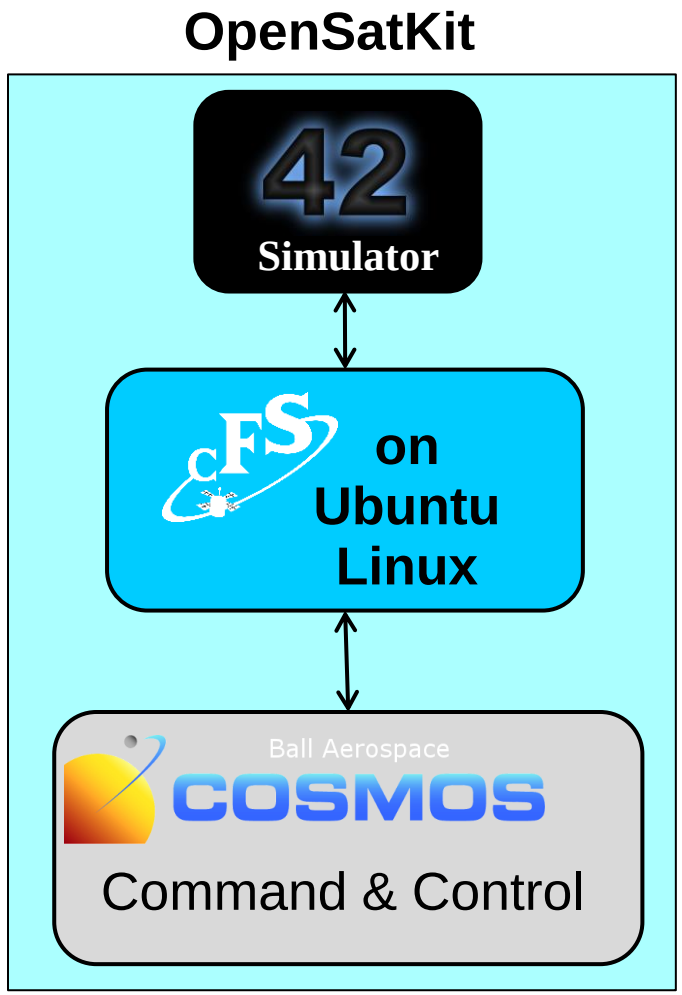
Workshop on Open-Source Space Missions Design Tools

David McComas
dmccomas63@gmail.com

3/1/22

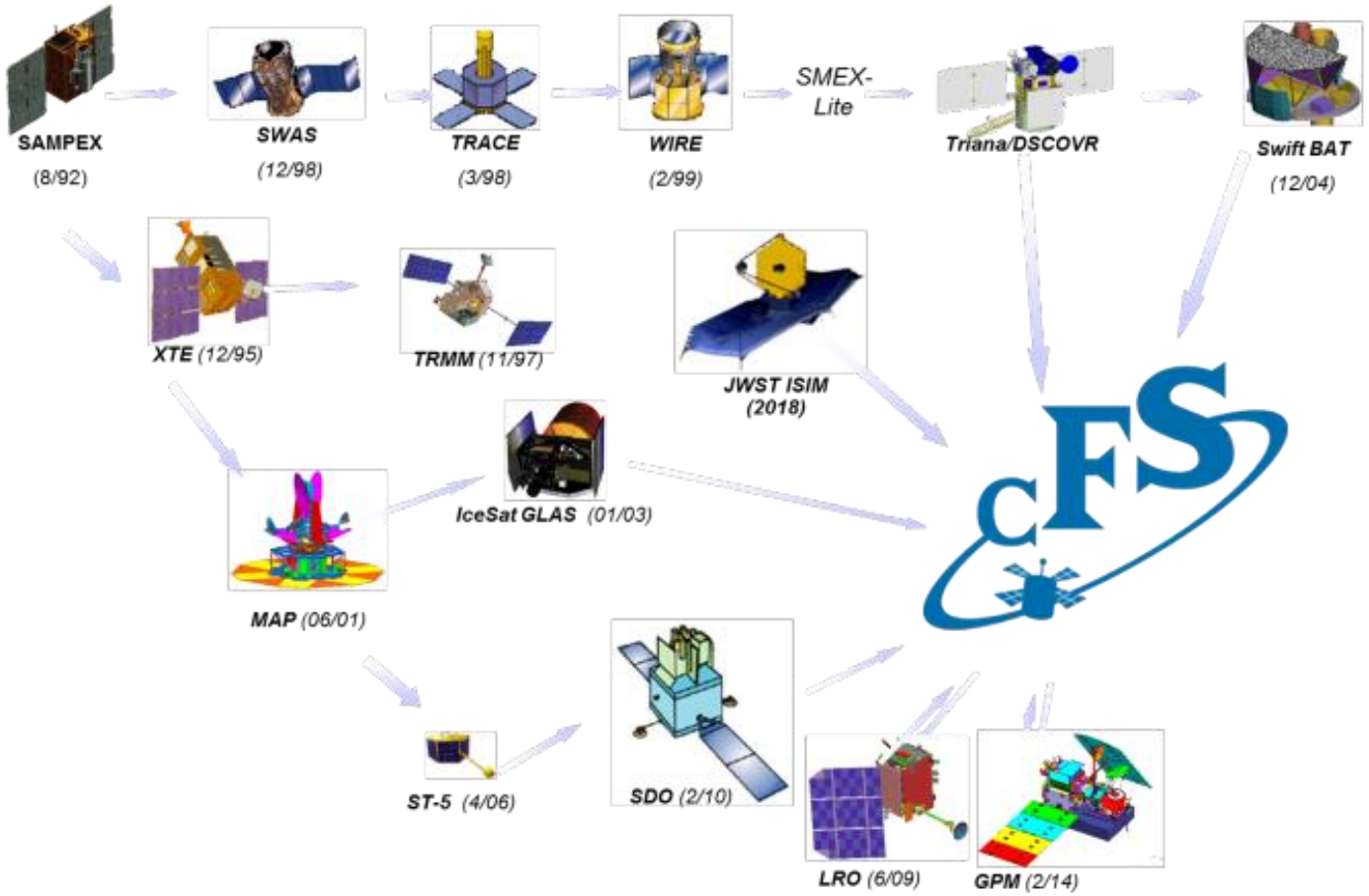
Open STEMware Foundation

Introduction



- **OpenSatKit was originally created to help flight software developers adopt and work with NASA's core Flight System (cFS)**
- **OpenSatKit combines COSMOS, cFS, and 42 into a single platform**
  - All users must install the complete system regardless of their goals
- **OpenSatKit is currently being decomposed into multiple components that can be combined to better serve user needs**
- **Agenda**
  1. Describe the new tool suite
  2. Demonstrate end-to-end simulation using the current OpenSatKit
  3. Demonstrate cFS app development using cFS Application Toolkit (cFSAT)
- **This slide deck contains many details that will not be presented, but will be valuable as a future reference**

NASA's core Flight System



NASA's core Flight System (cFS) is an open-source flight software (FSW) framework providing an application runtime environment that is portable across different hardware/operating system platforms.

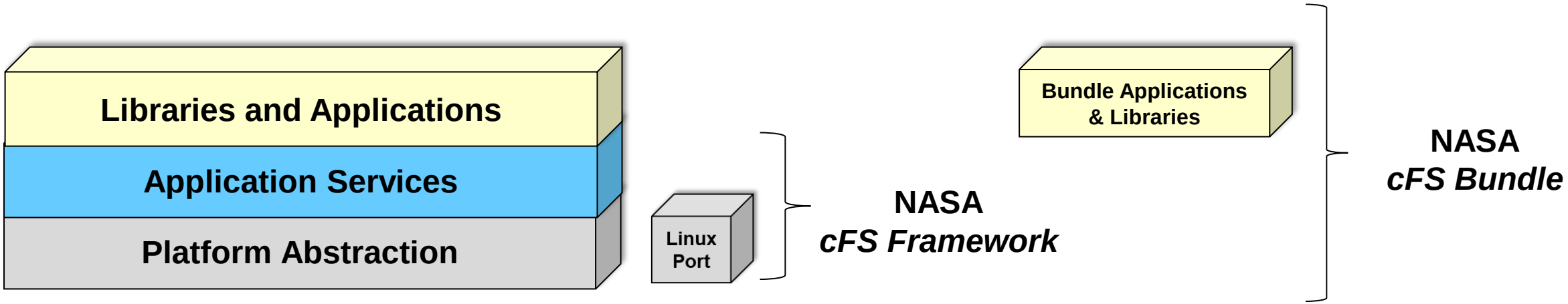
The cFS provides a high quality FSW with decades of flight heritage.

cFS Layered Architecture

Platform Abstraction: Provides portability across different operating systems including Linux, FreeRTOS, and RTEMS

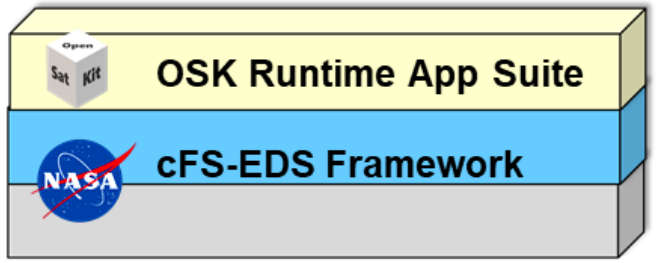
Application Services: Creates an application runtime environment that allows apps to be reused across projects

Libraries and Apps: Provide project functionality



OpenSatKit Overview

- **OpenSatKit (OSK) is suite of open-source products built upon NASA's core Flight System (cFS)**
- **It provides:**
  1. A cFS educational platform
  2. A design & development environment for mission flight software applications
  3. Platforms for STEM** education and hobbyists
  4. Platforms for technology development



OSK Runtime Bundle

- Built on NASA's cFS Electronic Data Sheet (EDS) Framework
  - EDS supports standards-based interface specifications
- The app suite provides essential flight software functionality
  - Command & Telemetry ground interfaces
  - Onboard file management and ground-flight file transfers
- Assured framework-app compatibility

**Science Technology Engineering and Mathematics

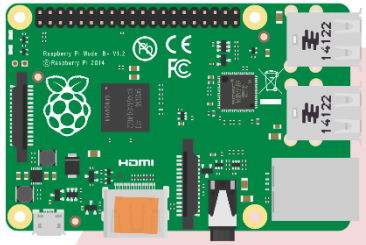
OpenSatKit Product Suite

PySimpleGUI
Command, Telemetry & Control

Ball Aerospace COSMOS
Command, Telemetry & Control

42 Simulator

cFSAT Bundle Adds:
- Educational Apps
- PySimpleGUI for Command, Telemetry, & Control

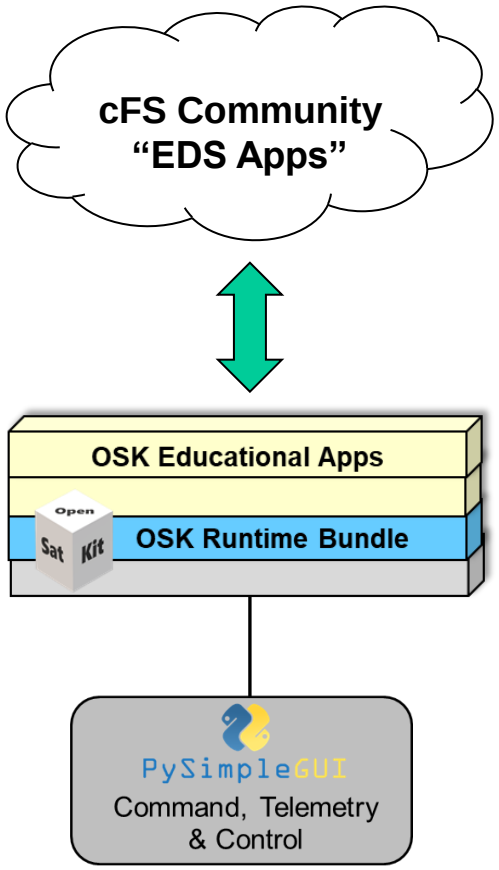


Pi-SAT Bundle Adds:
- Raspberry Pi Apps
- PySimpleGUI for Command, Telemetry, & Control

Simulation Bundle Adds:
- Mission Apps
- 42 Simulator
- COSMOS for Command, Telemetry, & Control

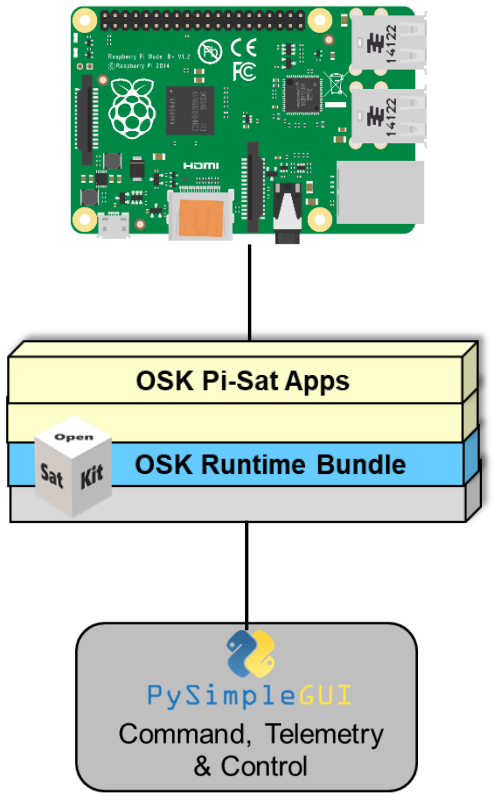
OpenSatKit Bundle
- Runtime App Suite
- cFS-EDS Framework

cFSAT



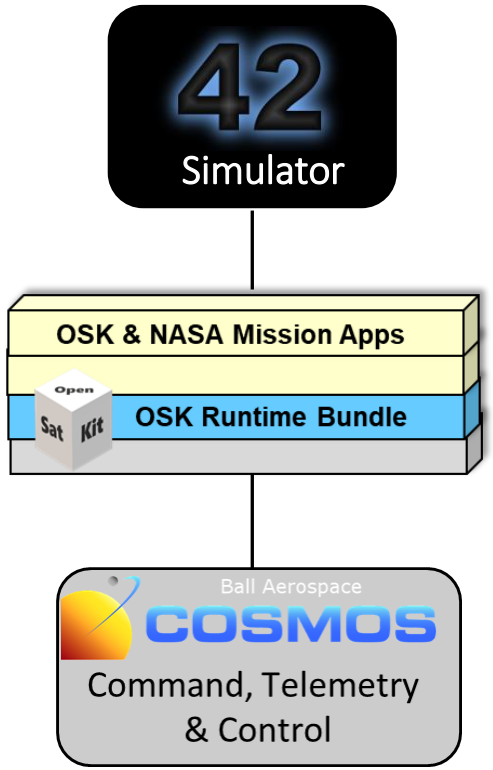
- **Lightweight platform that serves as a cFS**
  - Educational platform
  - Application development environment
- **Suitable for software STEM educational projects including**
  - FSW application development and test
  - Ground software data processing
- **Uses the cFS with Electronic Data Sheets (EDS) Framework**
  - Includes an EDS toolchain that creates ground and flight software artifacts from cFS apps packaged with an EDS interface specification
  - Uses latest cFS v7 Caelum release
- **Educational apps for training and self-guided tutorials**
- **Minimalistic Command, Telemetry and Control ground system**
  - Written in a lightweight portable python GUI framework called PySimpleGUI
- **OpenSatKit-Apps is a github repository of cFS Apps with EDS specifications**

Pi-SAT



- **Adds Raspberry Pi libraries and apps**
  - Use low-cost Raspberry Pi boards to learn and use the cFS Framework
- **Low-cost hardware platform ideal for learning how to write cFS *"interface apps"***
  - Provides command and telemetry interface to a hardware device
  - Use Python to talk with your system
- **Base platform can be customized for STEM educational projects**

cFSIM



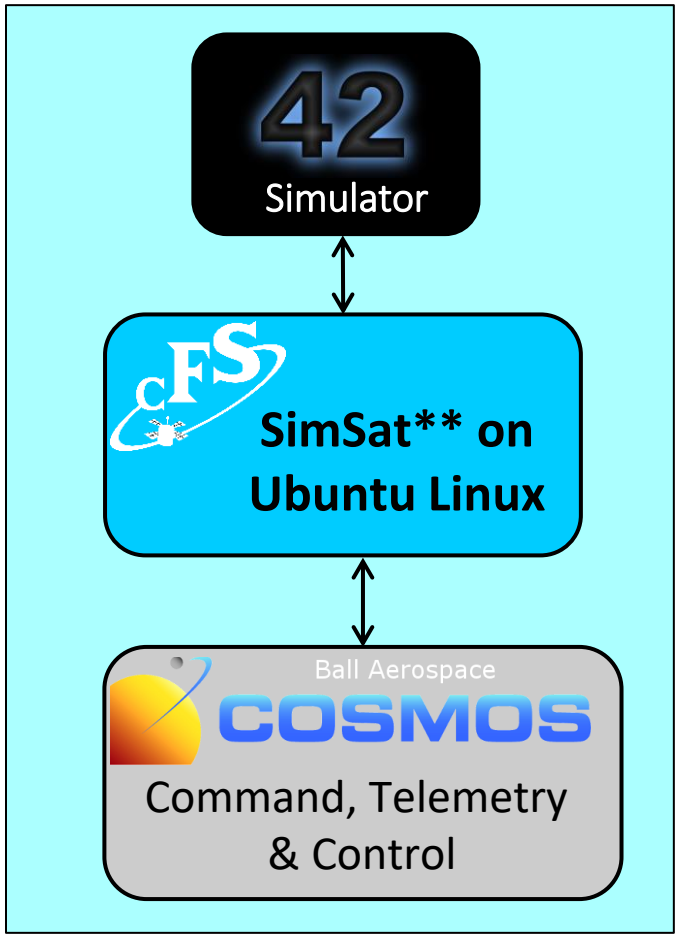
- **Complete ground, flight, and dynamic simulation system including**
  - Ball Aerospace's COSMOS command and control platform for embedded systems
  - NASA Goddard's 42 dynamic simulator

- **Suitable for**
  - Flight software application development and test
  - End-to-end software simulation
  - Ground software data processing

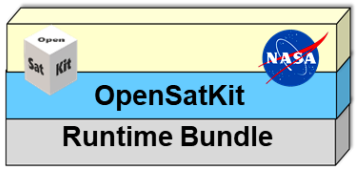
2022 OSK Product Roadmap

Current OpenSatKit

https://github.com/OpenSatKit/OpenSatKit/wiki



Refactor
&
Distribute



cFSIM

Beta Release

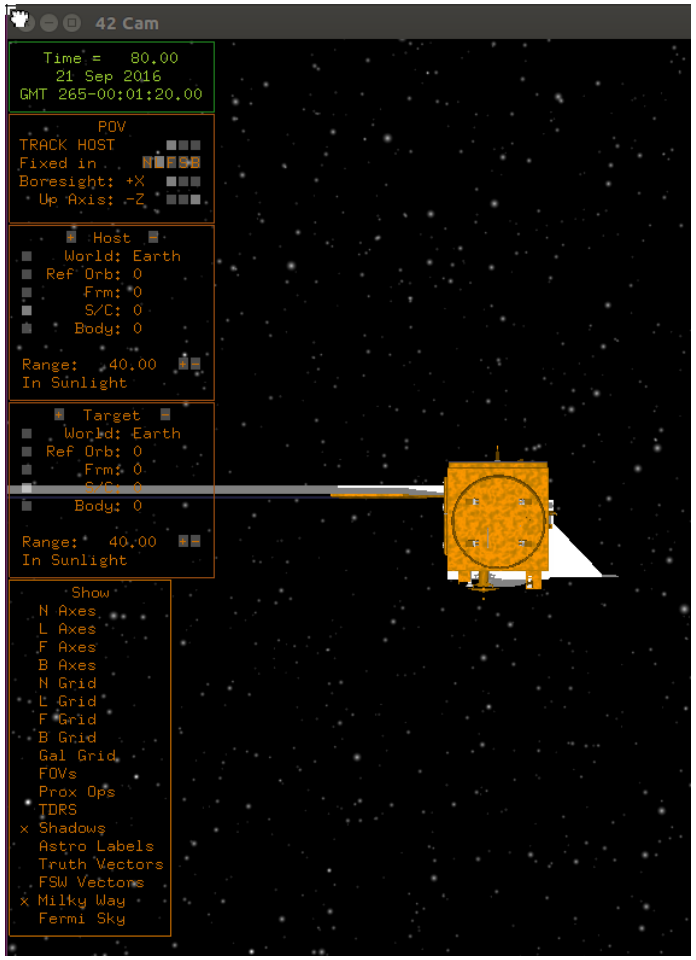
https://github.com/OpenSatKit/cfsat

cFSAT

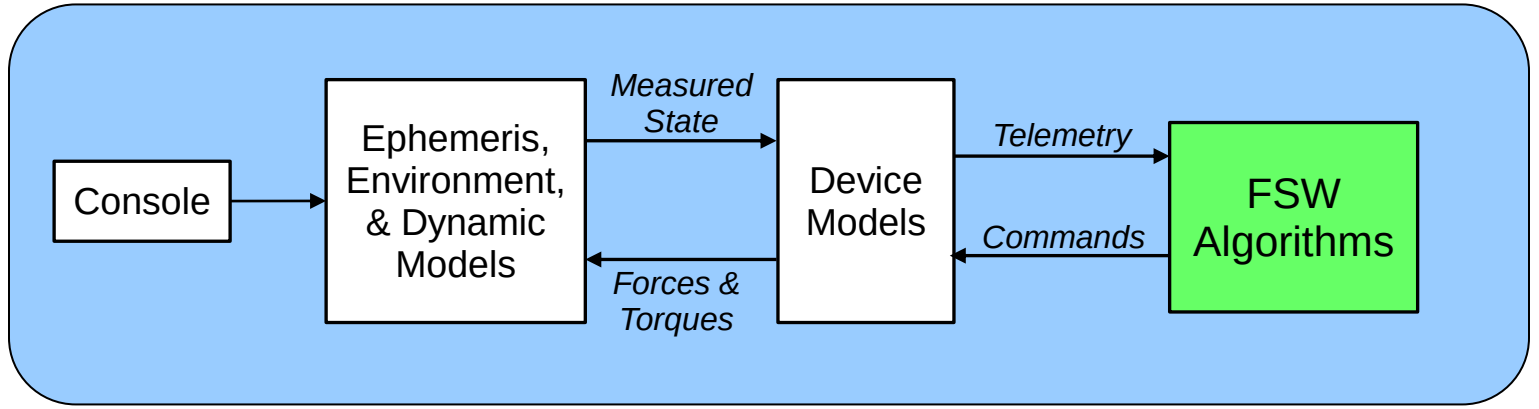
****SimSat** = Reference mission flight software app suite

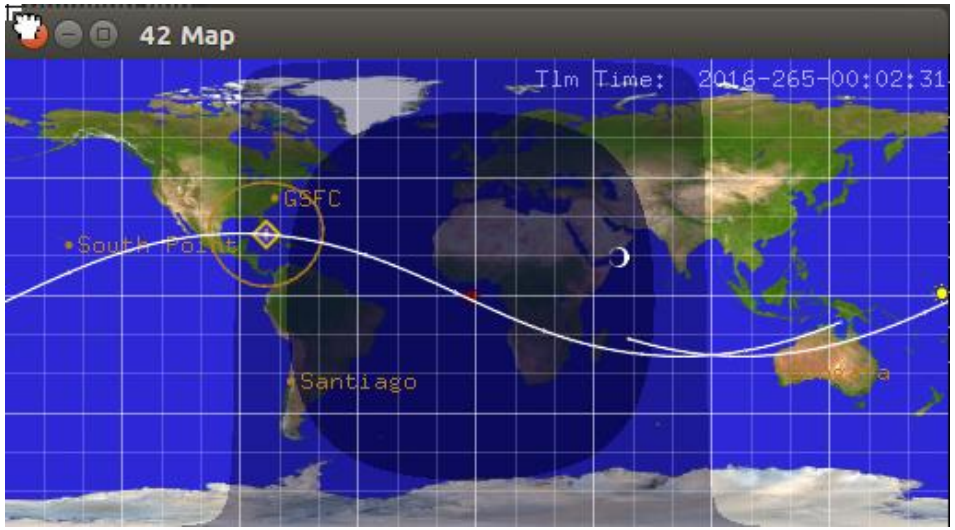

OpenSatKit
End-to-End Simulation

42 Standalone



42 Simulator



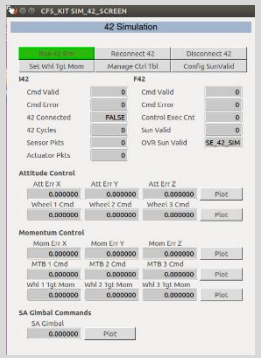


User interact with the simulation via the console and graphics

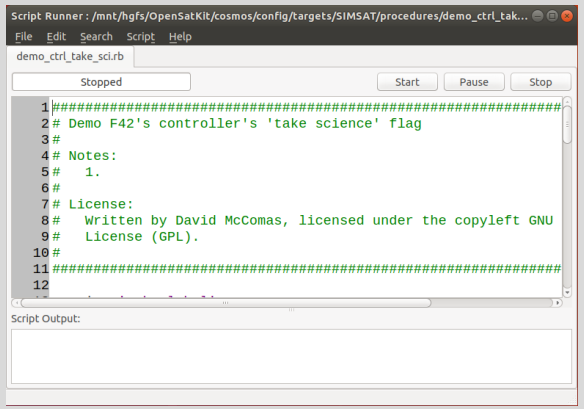
OpenSatKit 42 Simulation

COSMOS

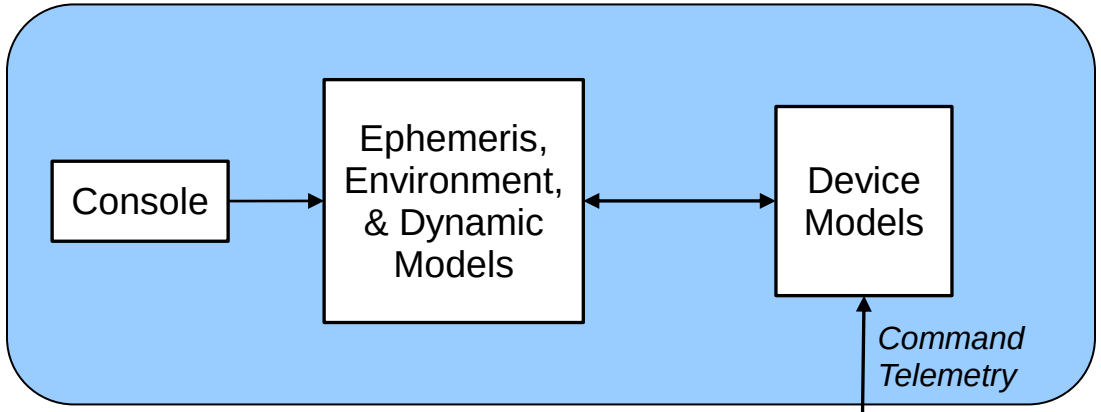
Screens



Scripts



42 Simulator



Flight Software algorithms Manually ported to an app

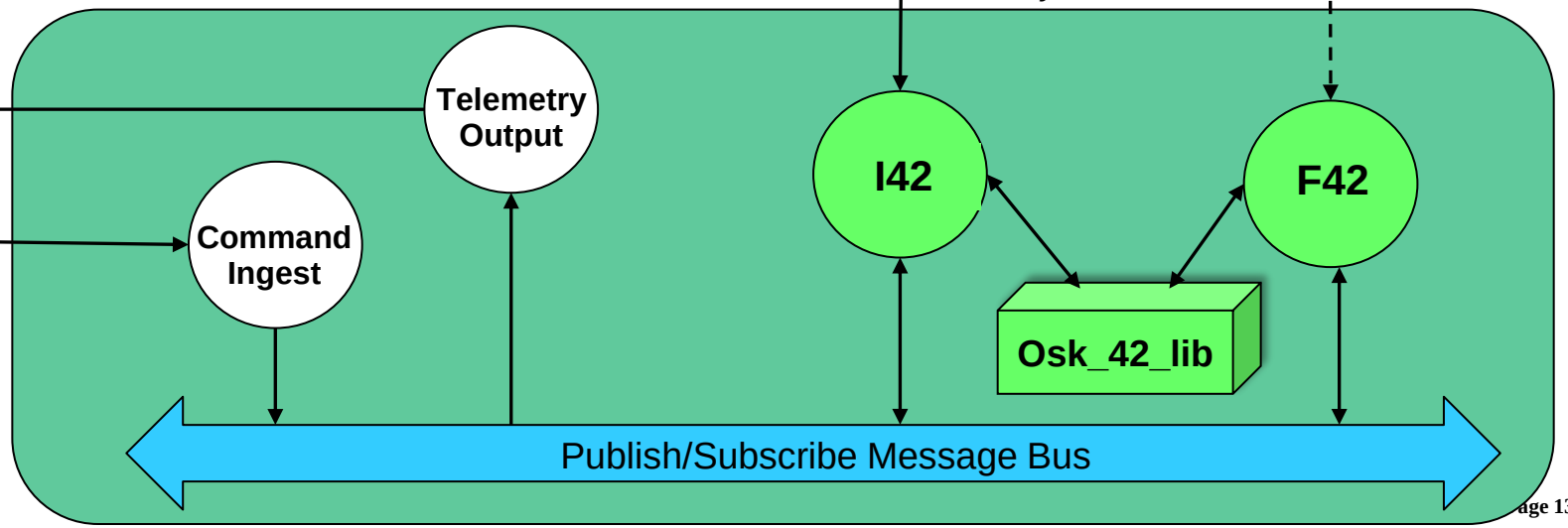
Socket Interface

Command
Telemetry

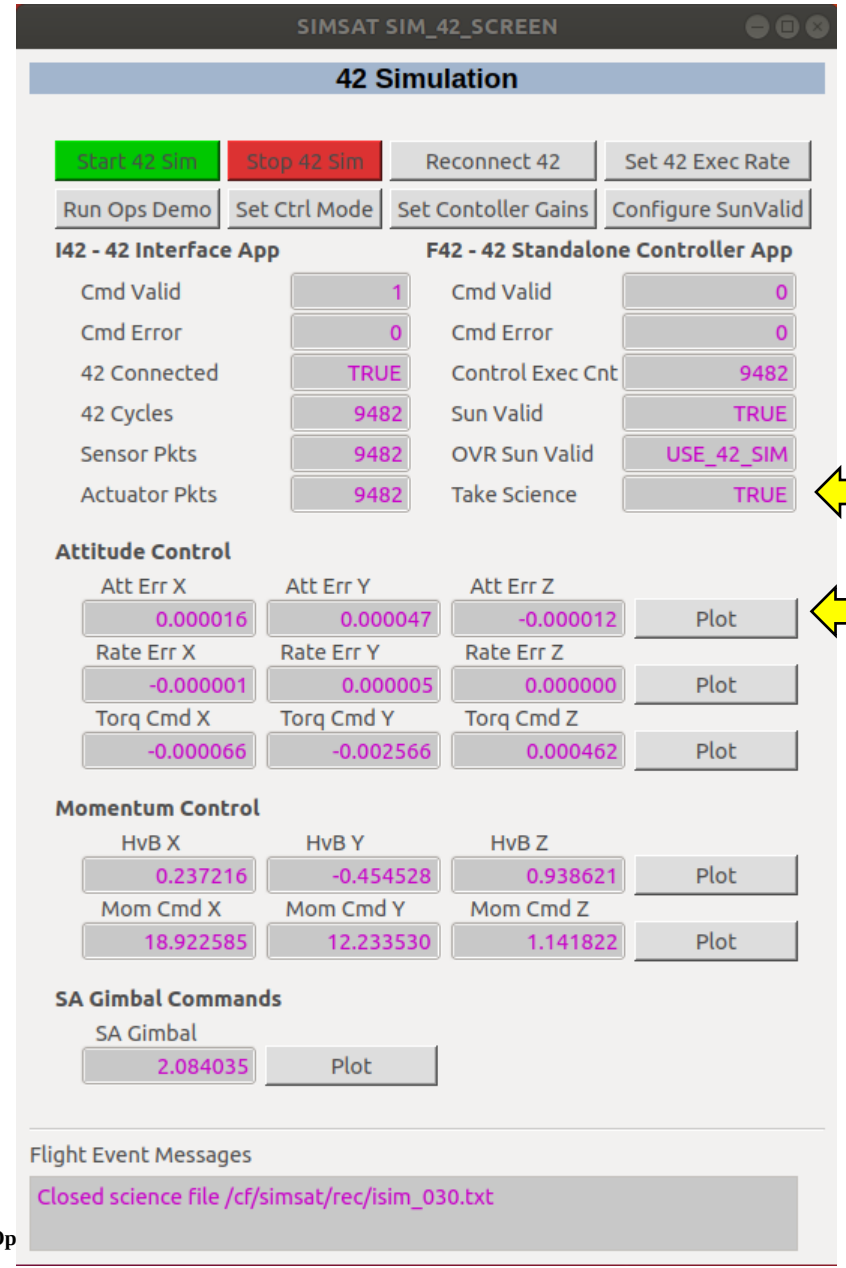
Flight Software

Telemetry

Commands



42 Flight Software App Telemetry



Flight software attitude control app defines a “Take Science” flag

- Used by “operations” to determine when to enable science data collection
- Set to TRUE when attitude errors on all three axes are below a threshold
- Default threshold is .0005 radians

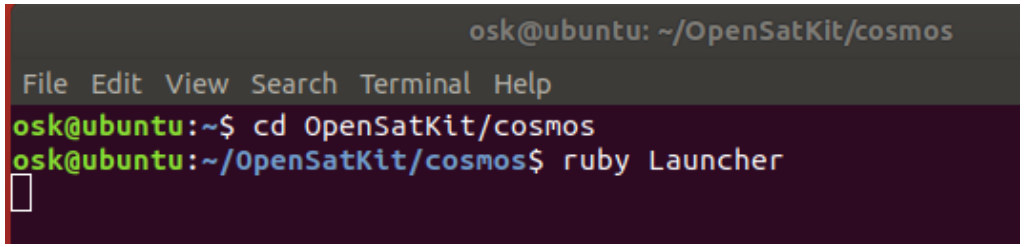
Demo Introduction

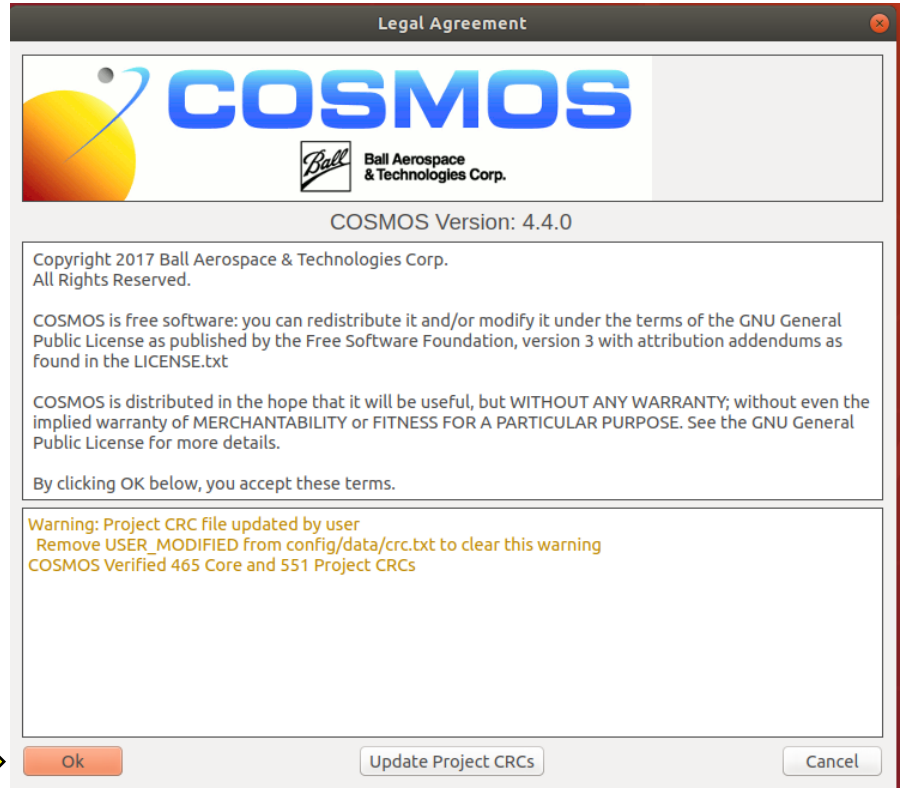
- **Demo 1**
  - Use OpenSatKit screens to run and interact with a simulation
  - Observe the behavior of the “take science” flag
- **Demo 2**
  - Run a script that manages an ops scenario
    - Waits for the “Take Science” flag to equal TRUE
    - Powers on a simulated science instrument
- **Notes**
  - OSK was originally designed for flight software developers so
  - OSK does not have the latest version of 42
  - The F42 app has changed with each 42 update, but not all F42’s functions have been updated
- **Please provide feedback!**
  - These demos can serve as the start of a conversation for how OSK could better serve students

Demo Reference Slides

Running OSK (1 of 2)

1. **Open a terminal window (Ctrl-Alt-t)**
2. **Navigate to the base directory where you installed OSK**
   - "~/" is used to indicate the OSK base directory so "~/cfs" is equivalent to "/home/user/OpenSatKit/cfs" if OpenSatKit was installed in the home directory for an account named "user"
3. **Change directory to cosmos**
   - **cd ~/cosmos**
4. **Start COSMOS**
   - **~/cosmos$ ruby Launcher**



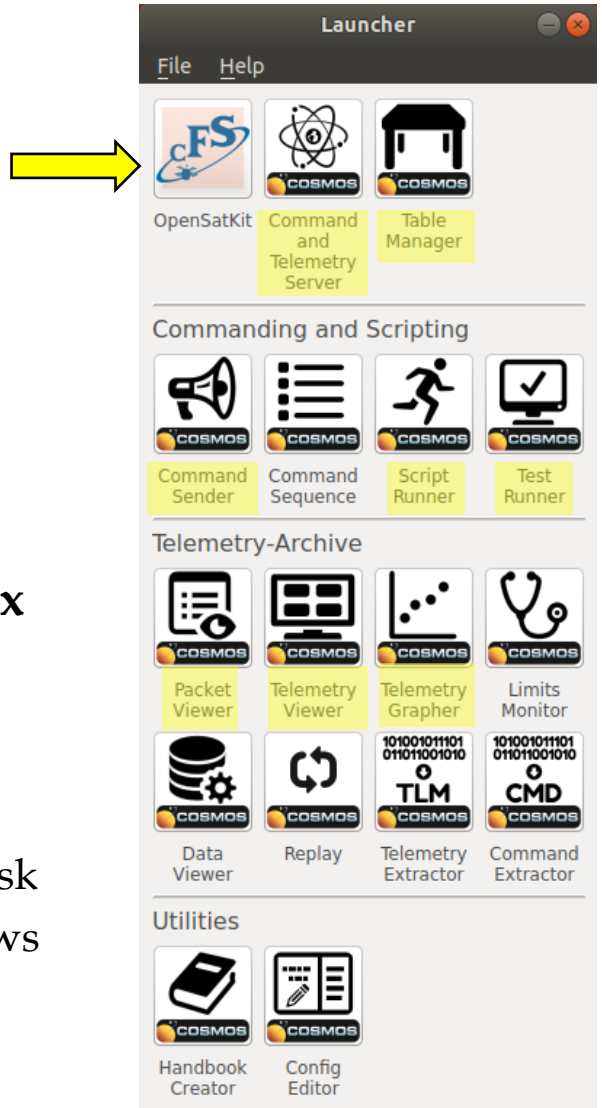


5. **Select <OK> to create the "Launcher" screen shown on the next slide**

Running OSK (2 of 2)

6. Select "OpenSatKit" icon with a single click

- This launches COSMOS's Command and Telemetry Server, Telemetry Viewer, and displays OSK's main window
- You can minimize the COSMOS tools, but don't close them



- **Each tools on the COSMOS "Launcher" runs as a separate Linux process with a Graphical User Interface (GUI)**
- **Shaded tool titles indicate the COSMOS tools used by OSK**
  - You do not have to invoke these tools directly
  - OSK screens launch COSMOS tools as they are needed to perform a task
  - A backup slide shows a COSMOS architectural view with the data flows between tools

OSK Screen Driven System

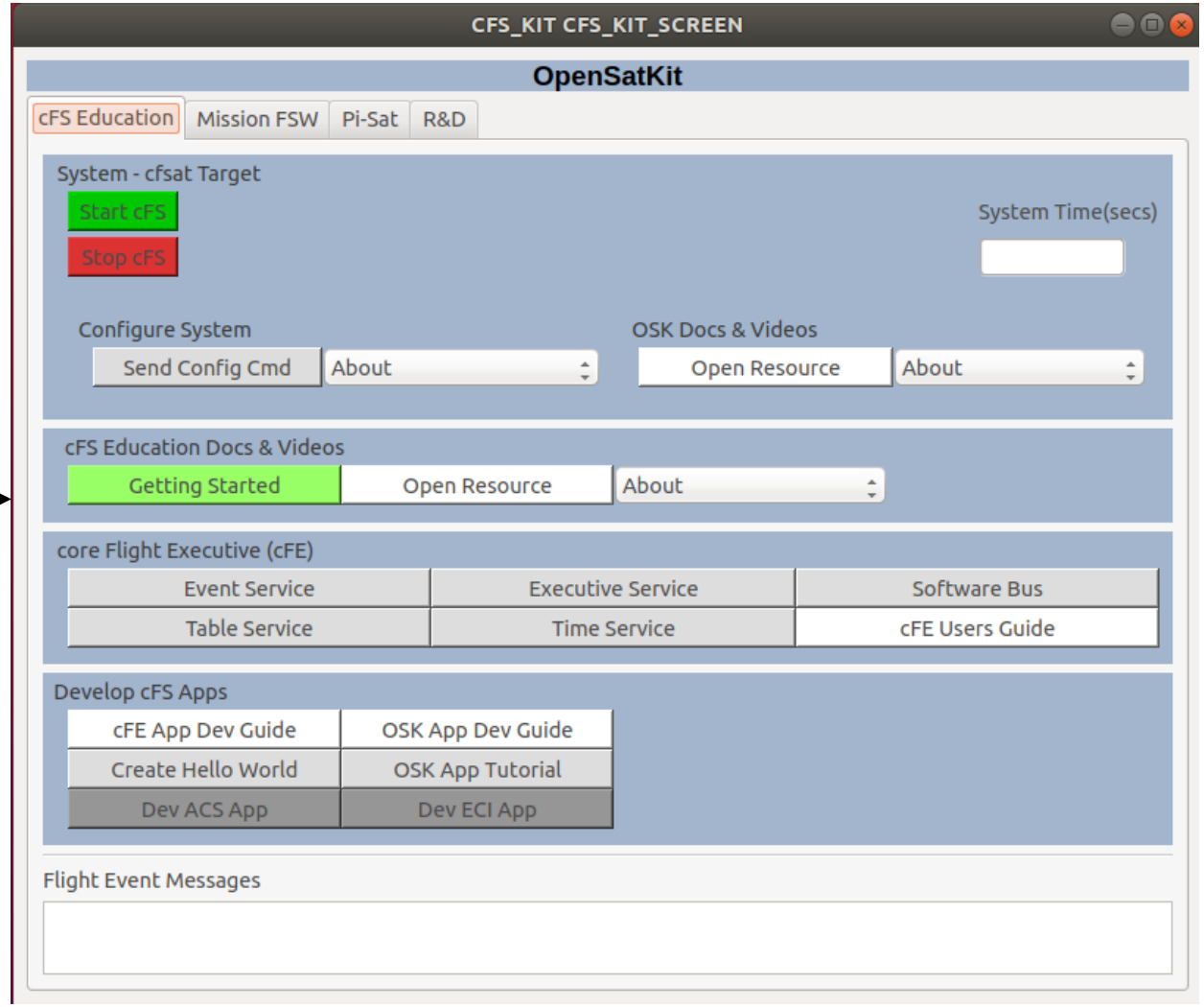
- Four tabs ***cFS Education***, **Mission FSW**, ***PiSat***, and ***R&D*** correspond to four cFS targets/ OSK Use Cases
- Each tab's screen has a similar layout and its own "Getting Started" Guide

User Objective Tab →

System/Target Management

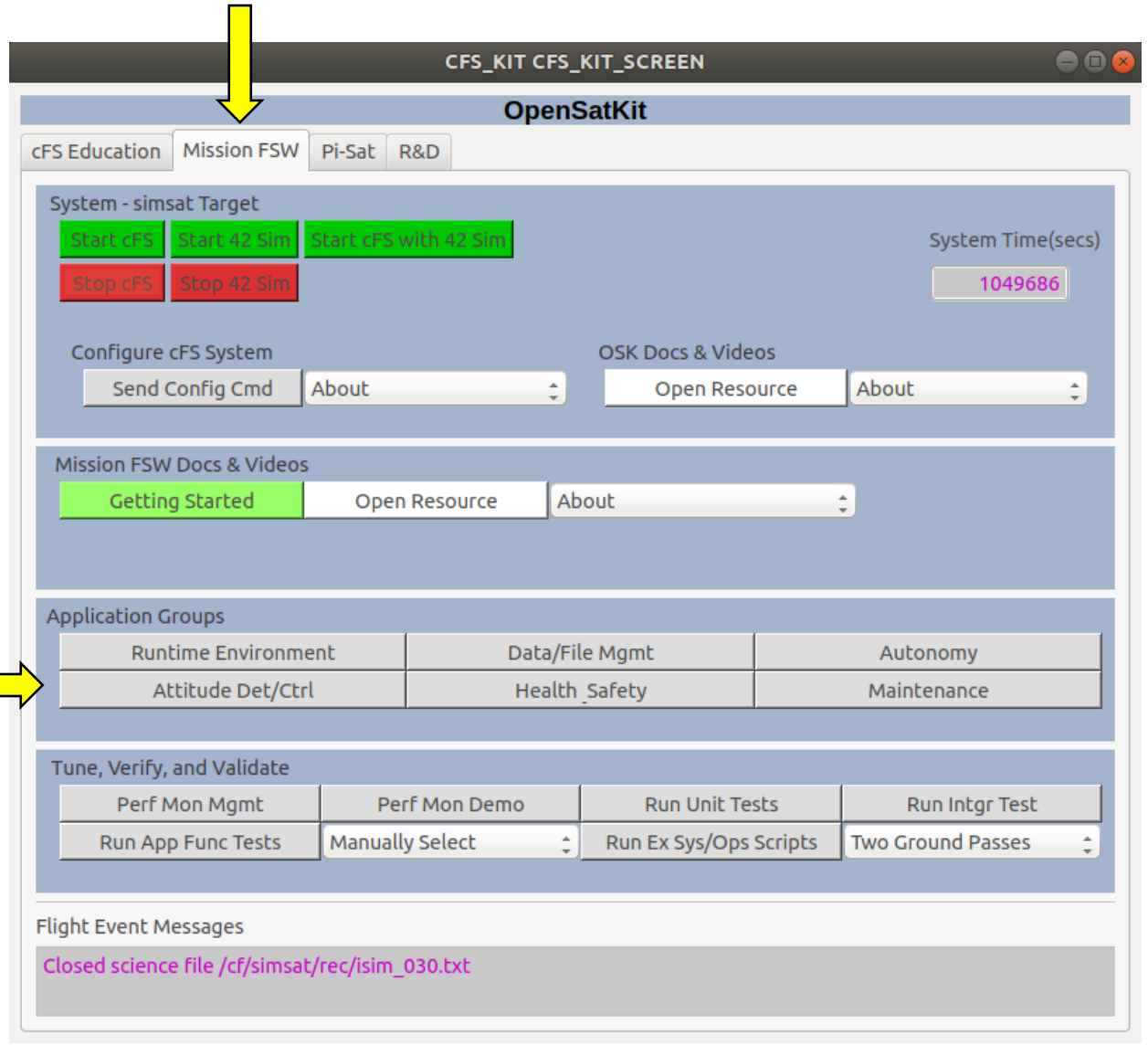
Learning Resources →

Content specific to *User Objective Tab*



Mission Flight Software Tab

The Mission Flight Software tab manages the end-to-end simulation

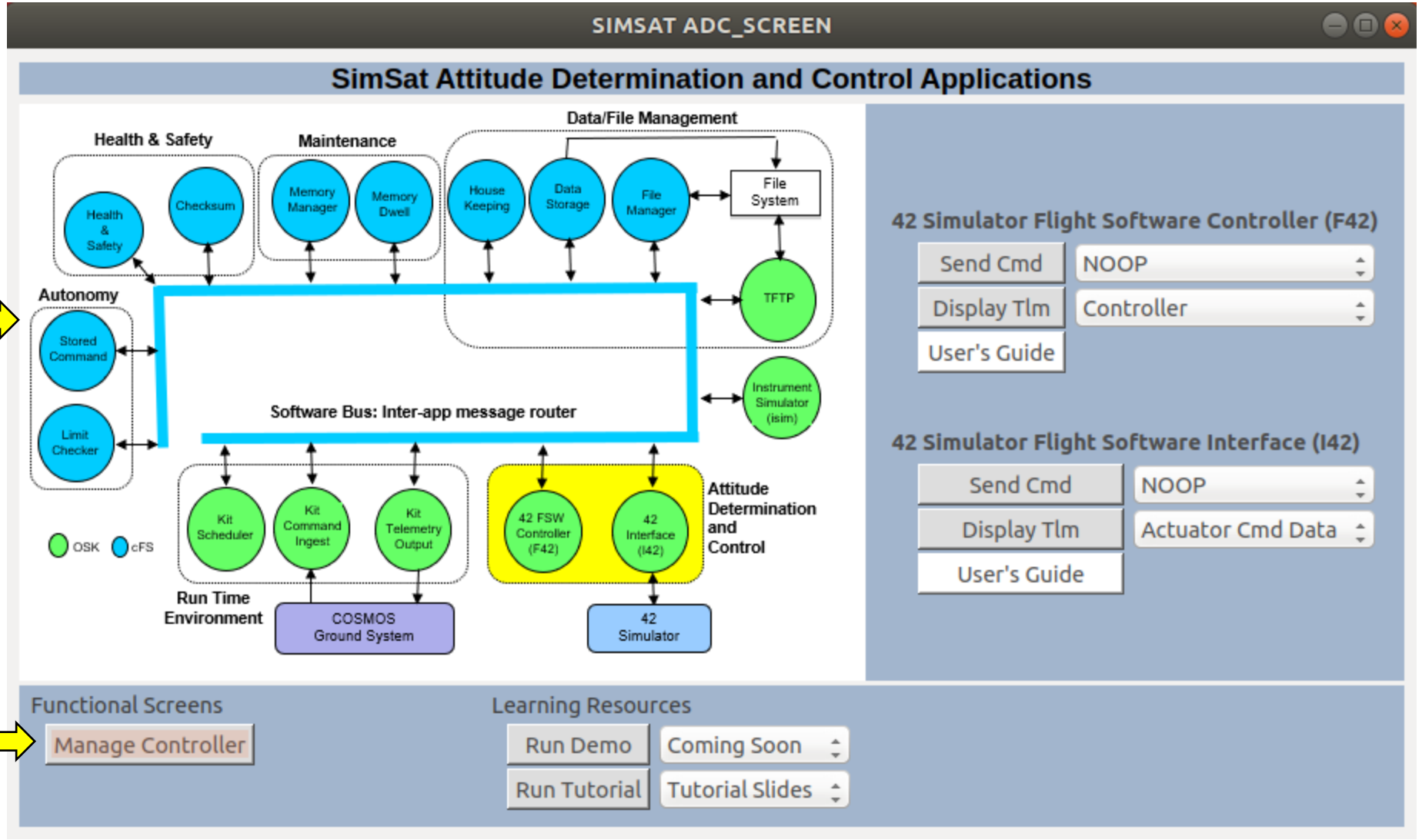


The attitude determination and control app group provides screens to interact with the I42 and F42 apps

Attitude Determination and Control Applications

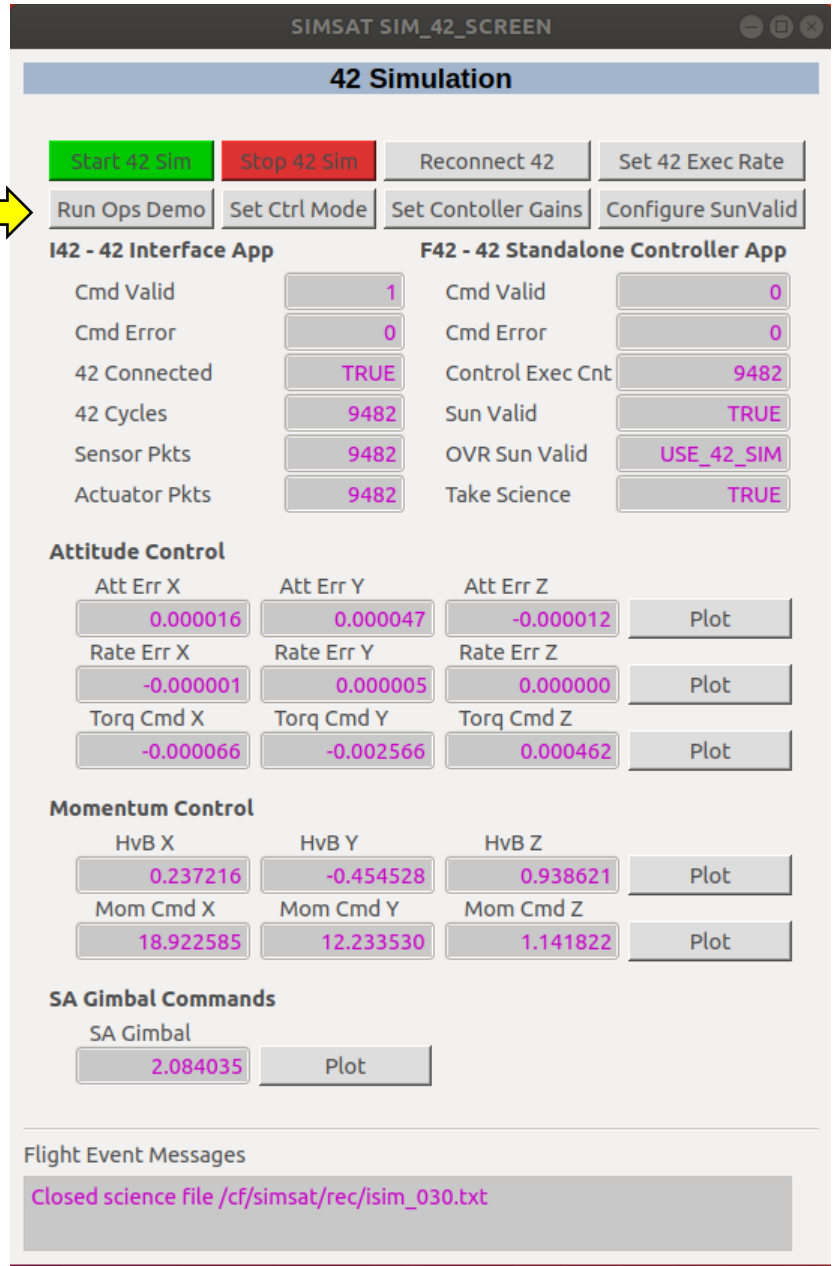
SimSat reference Mission app suite

Launch screen that is used in the demo



42 Flight Software App Telemetry

Start script-based Demo using COSMOS Script Runner



Ops Script Demo

Script Runner : /mnt/hgfs/OpenSatKit/cosmos/config/targets/SIMSAT/procedures/demo_ctrl_take_sci.rb

File Edit Search Script Help

demo_ctrl_take_sci.rb

Stopped | Start | Pause | Stop

```
1  ##############################################################################
2  # Demo F42's controller's 'take science' flag
3  #
4  # Notes:
5  #   1. This demo intentionally has a limited scope in order to minimize
6  #      complexity and to make it suitable for presentations
7  #
8  # License:
9  #   Written by David McComas, licensed under the copyleft GNU General Publi
10 #   License (GPL).
11 #
12 ##############################################################################
```

Script Output:

/mnt/hgfs/OpenSatKit/cosmos/config/targets/SIMSAT/procedures/demo_ctrl_take_sci.rb saved

OSK Document Roadmap

OpenSatKit/docs

1 – *OSK Quick Start*: Top-level introduction to OSK and a roadmap for more in-depth engagement

3 – *OSK COSMOS Guide*: Describes how COSMOS has been configured and extended for OSK

– *OSK App Developer's Guide*: Describes how to develop apps using the OSK application framework

OpenSatKit/cosmos/config/targets/CFSAT/docs (cFS educational platform)

2 – *cFS Education Quick Start Guide*: Introduction to OSK's cFS educational target and associated resources

– *core Flight System (cFS) Overview*: Introduction to flight software (FSW) and NASA's cFS

– *core Flight Executive (cFE) Overview*: Overview of the cFE framework and its application services

OpenSatKit/cosmos/config/targets/CFE_[service] /docs

– Each cFE service contains its own tutorial document

OpenSatKit/cosmos/config/targets/SIMSAT/docs (cFS-based mission)

– *Mission FSW Quick Start Guide*: Introduction to OSK's cFS-based mission target and associated resources

– *Simple Sat Overview*: Describes the SimpleSat reference mission

– *Application Group Guides*: Multiple documents that describe how groups of cFS community apps work together

OpenSatKit/cosmos/config/targets/PISAT/docs (Raspberry Pi distro for STEM education)

– *Pi-Sat Quick Start Guide*: Introduction to OSK's Raspberry Pi target and associated resources

OpenSatKit/cosmos/config/targets/SANDBOX/docs (cFS application playground)

– *Research & Development Quick Start Guide*: Introduction to OSK's R&D target and associated resources

Recommended reading order if you're new to the cFS. The next steps depends on your goals.

OpenSatKit 42 Notes

- **42 configuration**
  - The *OpenSatKit/42/OSK* directory contains the 42 configuration files used in the simulation
  - Flight software and 42 time is not synchronized
- **Simple/Simulated Satellite (SimSat) is a fictitious reference mission**
  - The SimSat Quick Start Guide is incomplete
- **The demo ops script is located at**
  - *OpenSatKit/cosmos/config/targets/SIMSAT/procedures/demo_ctrl_take_sci.rb*
- **I42 and F42 command and telemetry definitions serve as the current documentation**
  - OpenSatKit/cosmos/config/targets/F42/cmd_tlm
  - OpenSatKit/cosmos/config/targets/I42/cmd_tlm
- **The process to develop and port new algorithms from 42 to a flight software app is complicated and undocumented**

Creating a “Hello World” Flight Software Application

Objectives

- **Guide participants through the process of creating and running a core Flight System (cFS) application**

- **Introduce technical concepts including**
  - cFS framework and applications
  - CCSDS Electronic Data Sheets (EDS)
  - OpenSatKit application framework

Approach

- **The short introduction will be a normal slide presentation**
- **After cFSAT is installed, we will run the python ground system and use it to launch tutorials that will step us through exercises**
- **We will perform each exercise together and I will pause for any questions**
- **Significant effort was made to minimize prerequisites**
  - Multiple concepts, systems, tools, and workflows are applied, but a detailed knowledge of these is not required
- **Versioning notes**
  - cFS is a prerelease of Caelum
  - cFSAT is a beta release
  - The OSK framework library (osk_c_fw) and demo app (osk_c_demo) originated from OSK 3.2 (cFS Aquila)
    - They have been updated to cFS Caelum
    - Transferred to their own repos in the OpenSatKit-Apps project

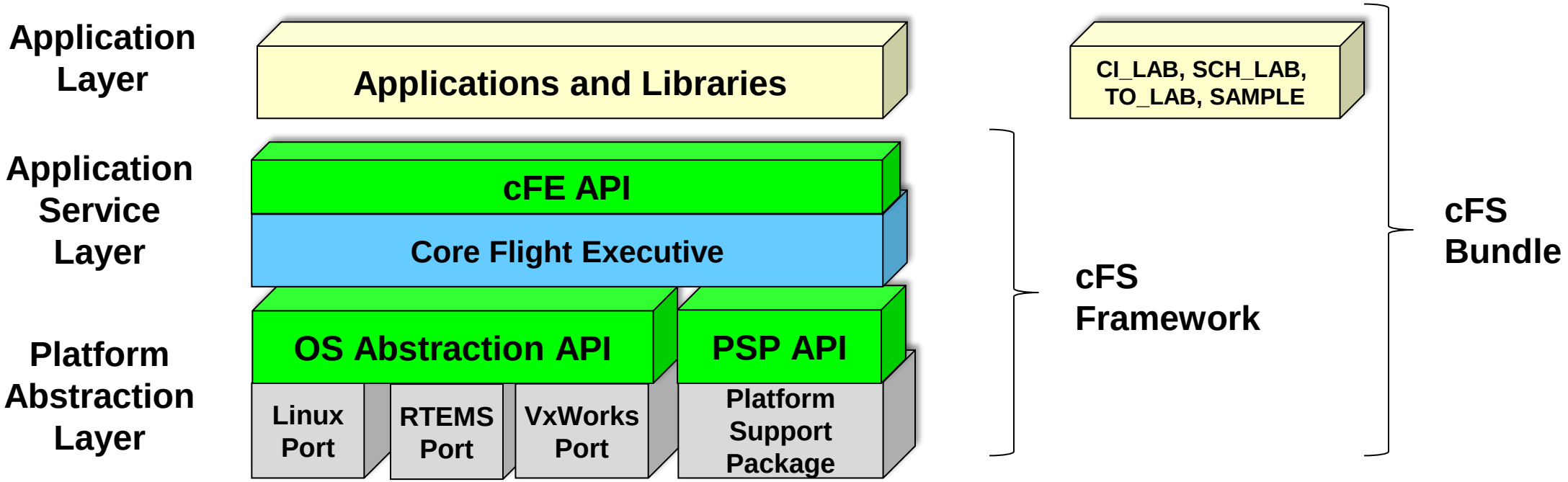
Agenda

1. **cFS and cFSAT Introduction**
2. **cFSAT Installation****
3. **Hands on Exercises**
   A. Build and Run the cFS
   B. Create Hello World App
   C. Modify Hello World App

** https://github.com/OpenSatKit/cfsat

cFS Architectural Overview

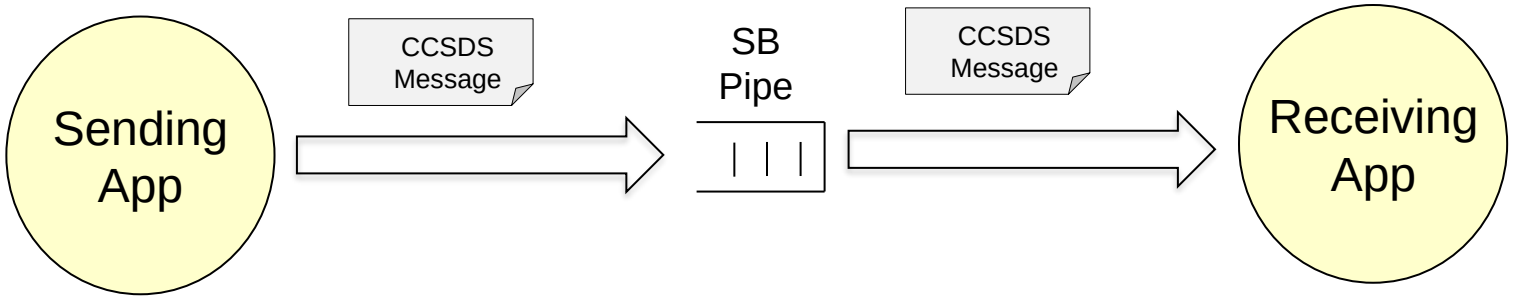
- **cFS uses a three-tiered software architecture that provides a portable flight software framework with a product line deployment model**
  - Platform Abstraction Layer ports to different operating systems (OS) / processor combinations
  - Compile-time configuration parameters and run-time command/table parameters provide adaptability and scalability
- **cFE Framework provides portable application runtime environment**
  - Mission functionality implemented by a combination of reusable and mission-specific apps



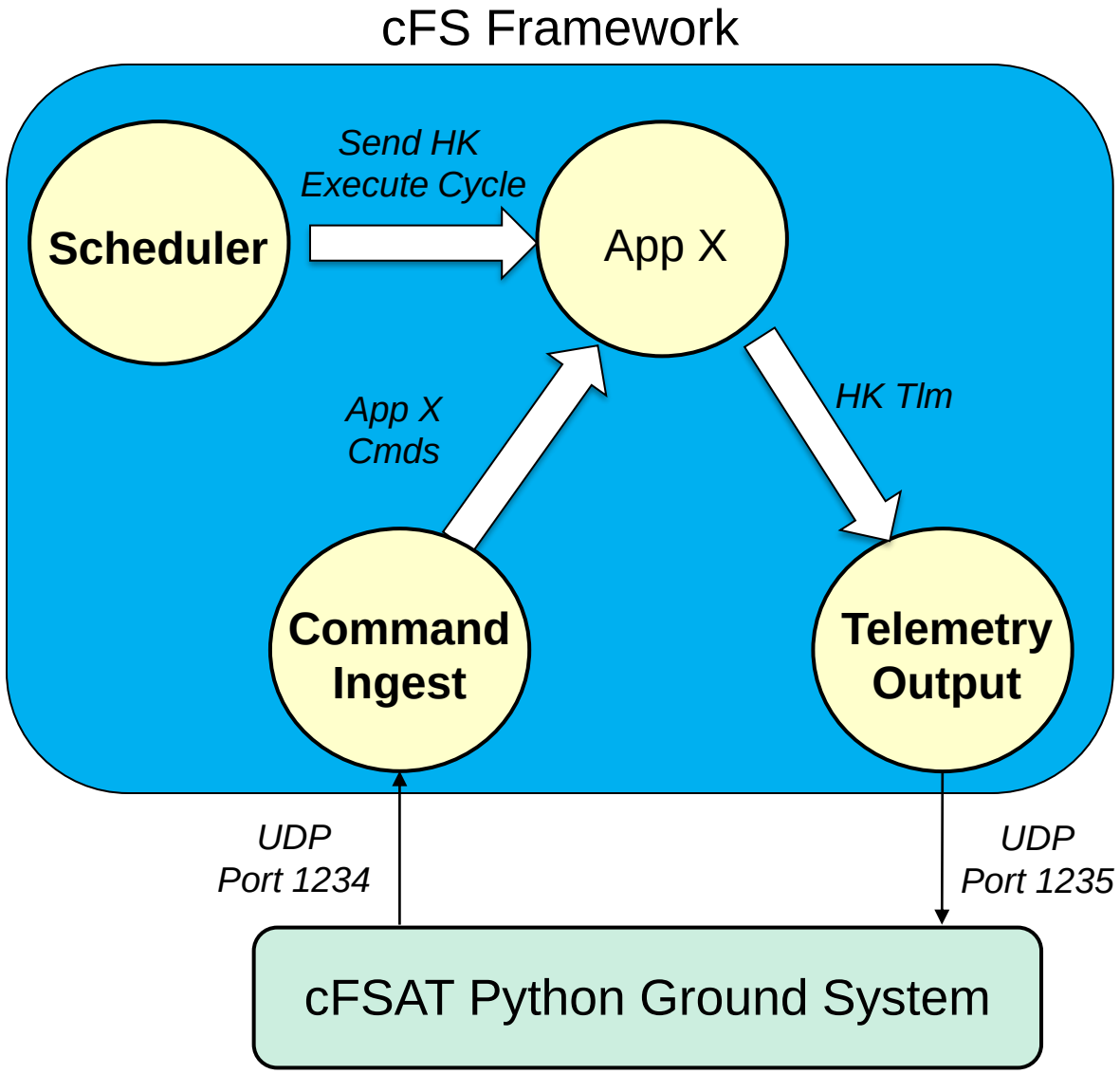
See backup slide for GitHub repos

cFS Application Architectural Context

- **The cFE, Operating System Abstraction Layer (OSAL), and Platform Support Package (PSP) provide services and access to system resources for applications**
- **The Software Bus (SB) is the predominant service that needs to be understood for this tutorial**
  - OSK's cfsat target documentation provides cFS educational material https://github.com/OpenSatKit/OpenSatKit/tree/master/cosmos/config/targets/CFSAT/docs
- **SB provides a publish/subscribe message bus**
  - Apps publish messages on the bus using a broadcast model
  - Apps received messages by creating pipes (FIFO queue) and subscribing to messages on a pipe



cFS Application Runtime Environment



A core set of apps are required to provide a runtime environment

- Different app implementations can provide customized solutions for different platforms
- File management & transfer not shown

Scheduler (SCH) sends messages at fixed time intervals to signal apps to perform a particular function

Command Ingest (CI) receives commands from an external source and publishes them on the SB

Telemetry Out (TO) receives messages from the SB and sends them to an external destination

Application Run Loop Messaging Example

Suspend execution until a message arrives on app's pipe

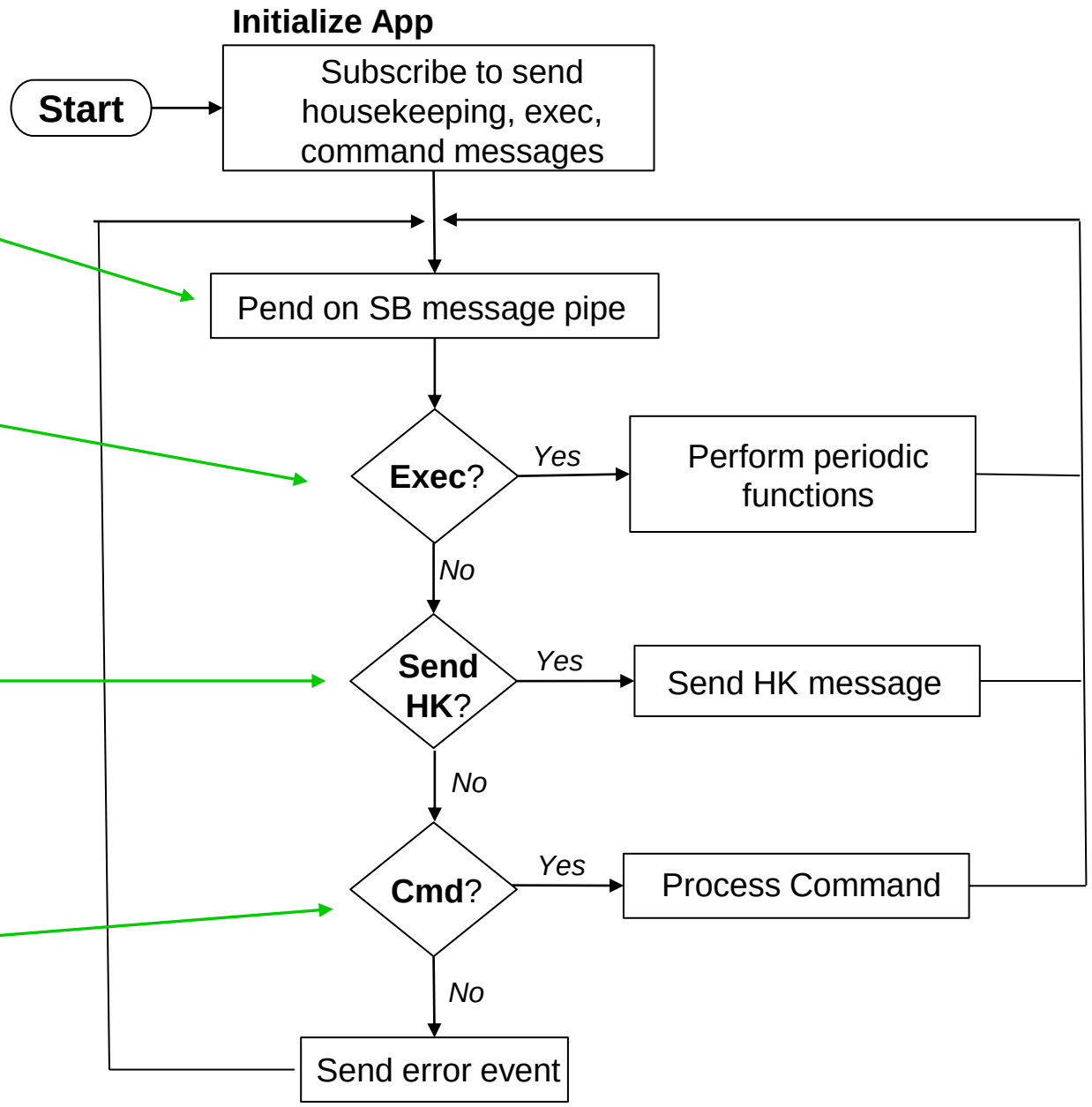
Periodic *execute* message from SCH app

Periodic *send housekeeping* message from SCH app

- Typically, on the order of seconds
- "Housekeeping cycle" convenient time to perform non-critical functions

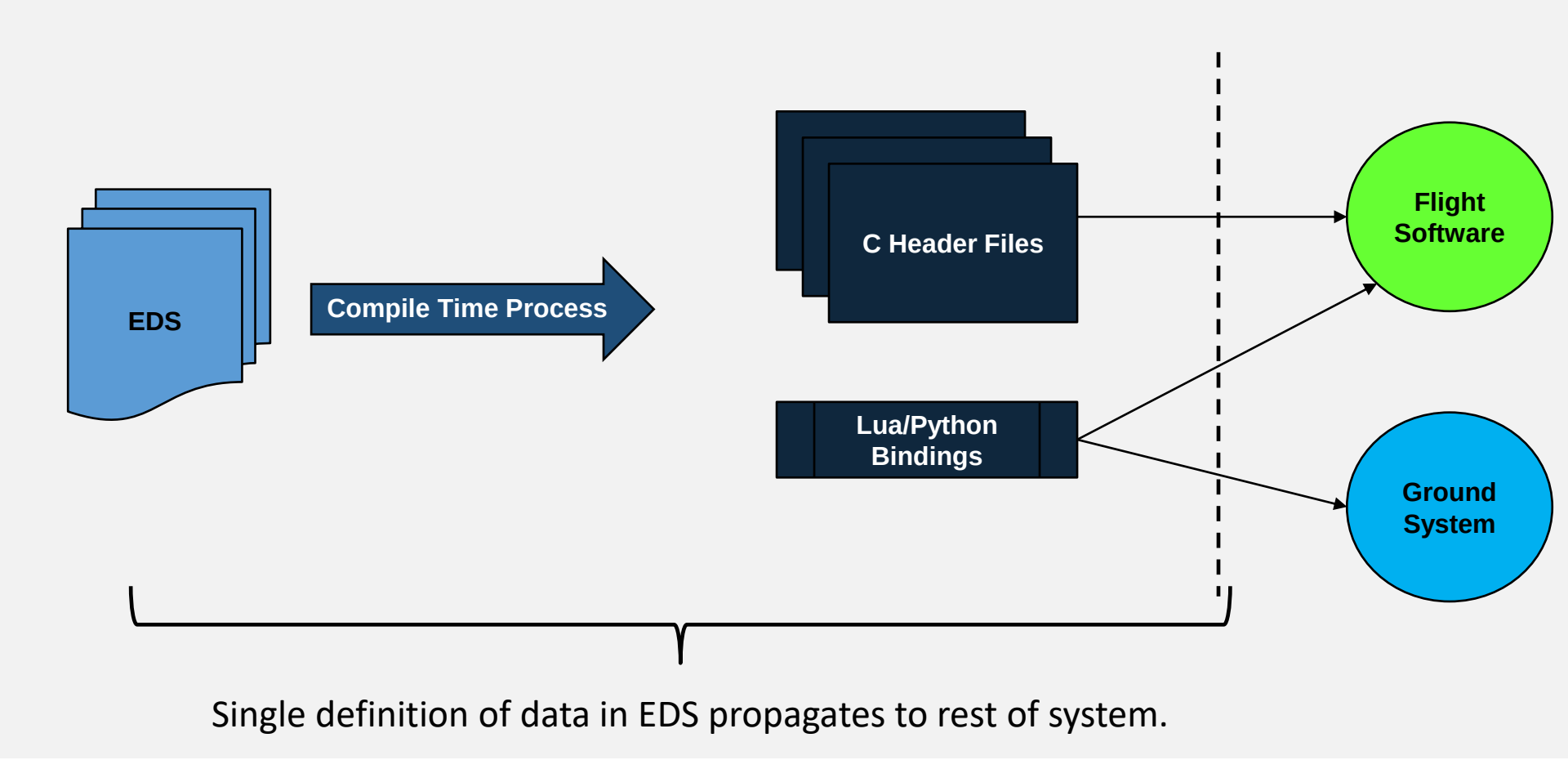
Process commands

- Commands can originate from ground or other onboard apps



Electronic Data Sheets

- **An Electronic Data Sheet (EDS) is a formal specification of a device, system, or software interface in a machine-readable format**
- **EDS specifies black box view of interfaces**



cFSAT Python Ground System

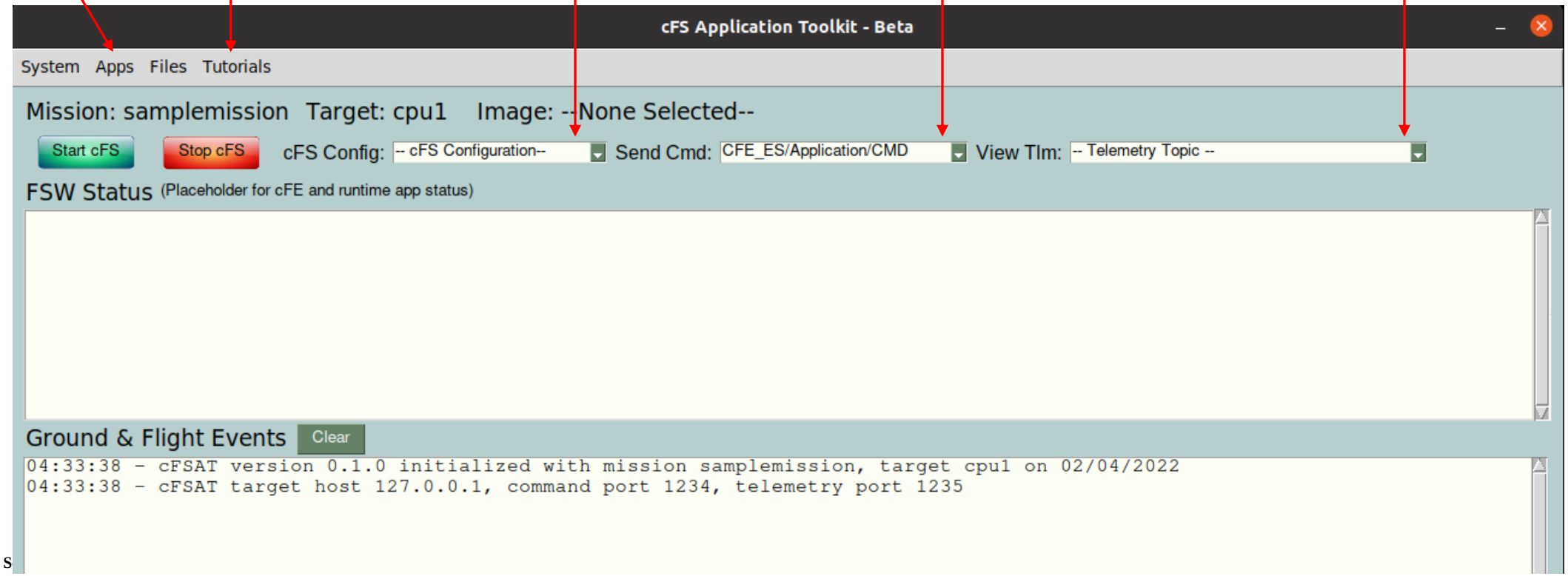
Create App Tool

Run Tutorial

Send Configuration Commands

Send Commands to cFS

Display Telemetry from cFS



Install cFSAT Prerequisites

Prerequisites

The system can be developed on any GNU/Linux development host. The following commands install the development packages for a Debian/Ubuntu environment. Other Linux distributions should provide a similar set of packages but, the package names and installation tool names may vary.

```
sudo apt-get update -y
sudo apt-get install build-essential
sudo apt-get install cmake
sudo apt-get install libexpat1-dev
sudo apt-get install liblua5.3-dev
sudo apt-get install libjson-c-dev
sudo apt-get install python3-dev
sudo apt-get install python3-pip
sudo apt-get install python3-tk
```

Package Notes:

- *sudo apt-get update* updates a platform's current package respositories
- *build-essential* contains a C developer tool suite including gcc, libc-dev, make, etc.*
- *cmake* must be at least v2.8.12
- *liblua5.3-dev* must be at least v5.1

The python appplication uses PySimpleGUI which can be installed with the following command:

```
pip3 install PySimpleGUI
```

Clone cFSAT Repo and Run Python Ground System

Clone cFSAT Repository

```
git clone https://github.com/OpenSatKit/cfsat.git
```

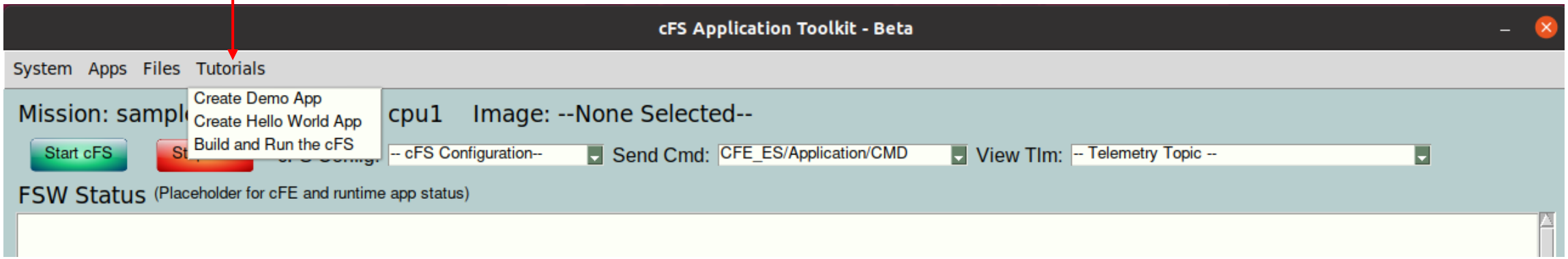
Run Python Ground System Applcation

In a new terminal window, starting in the directory where you issued the git clone, run the Ground System application and establish telemetry flow:

```
cd cfsat/gnd-sys/app
. ./setvars.sh
python3 cfsat.py
```

Hand on Exercises

The remainder of this workshop will be performed by launching tutorials from cFSAT's Tutorial menu



We will be using the following tutorials

1. Build and Run the cFS
2. Create Hello World App