



# Getting Started with OSK's Mission Flight Software

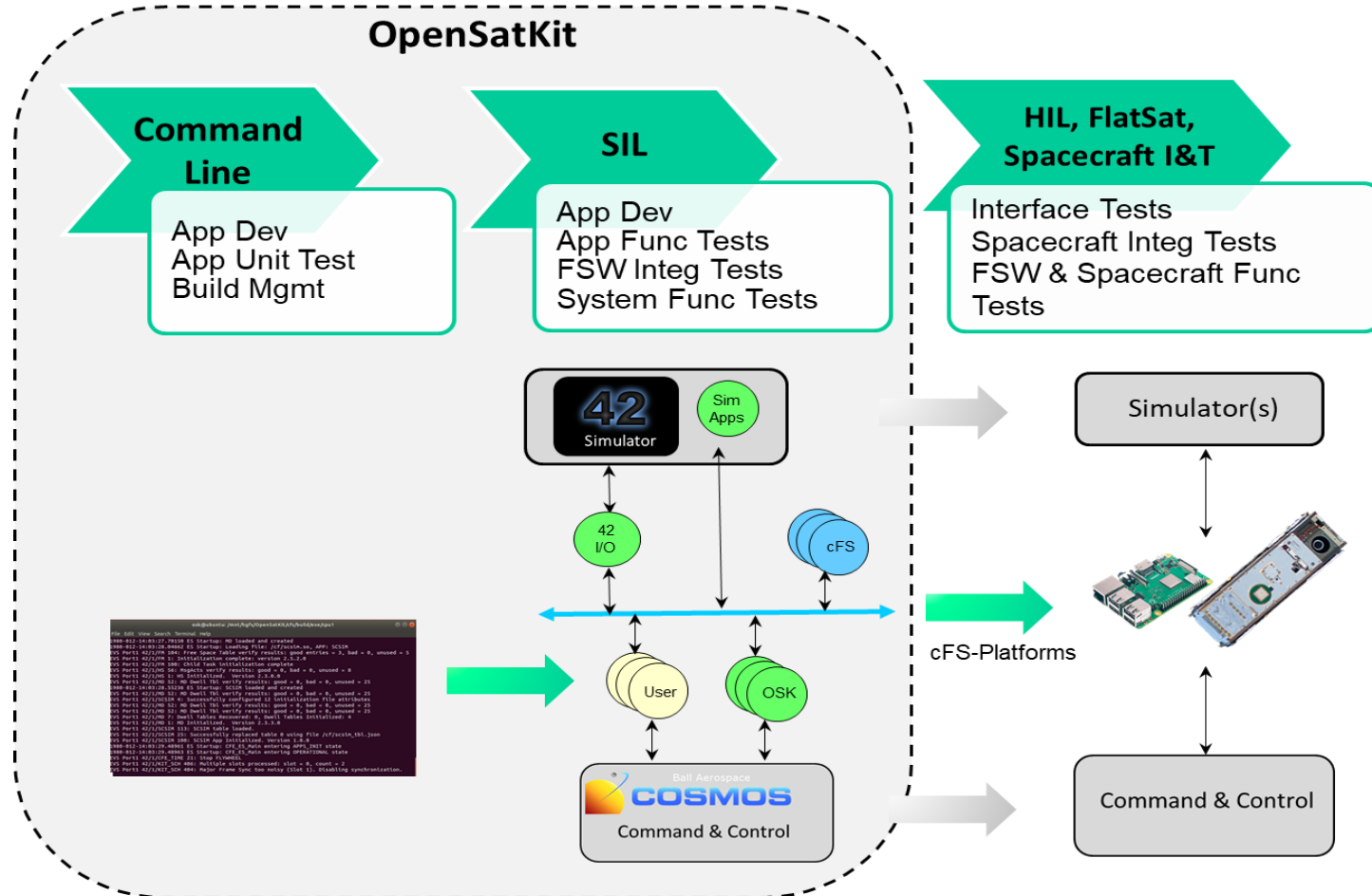
These are a collection of notes that will be transformed into a document in OSK v3.2

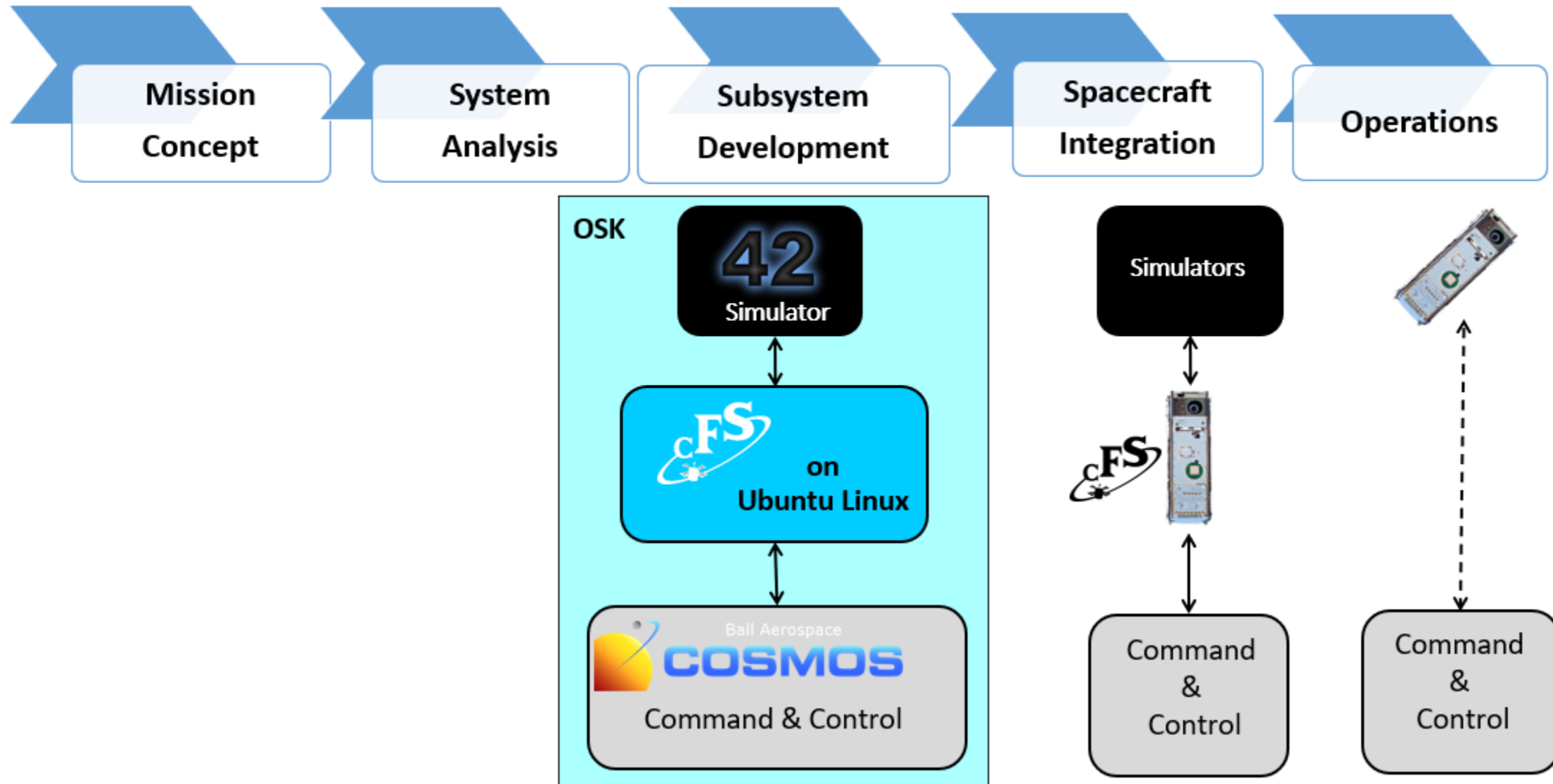
OSK v3.1



- 1. System Engineering Introduction**
- 2. Apply System Engineering Development Processes**
- 3. Apply System Engineering V&V Processes**

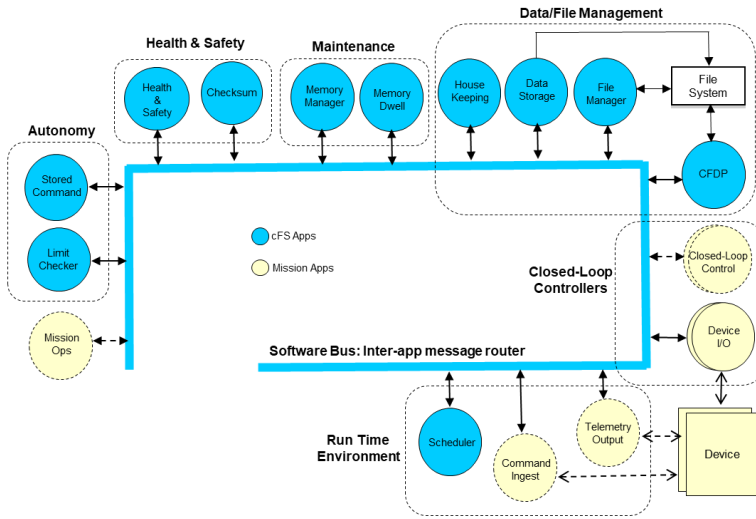
## Spacecraft FSW development phases







## Generic Spacecraft Model

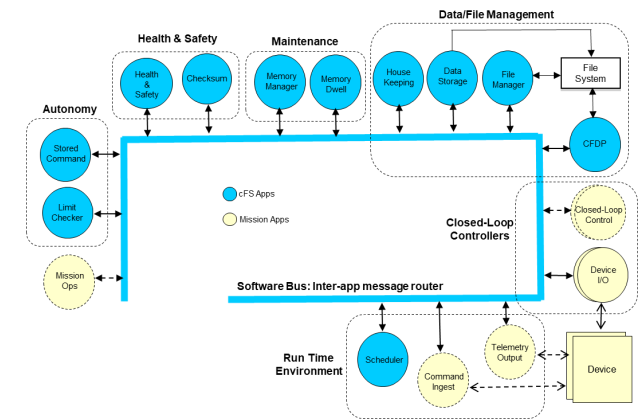


## Mission Inputs

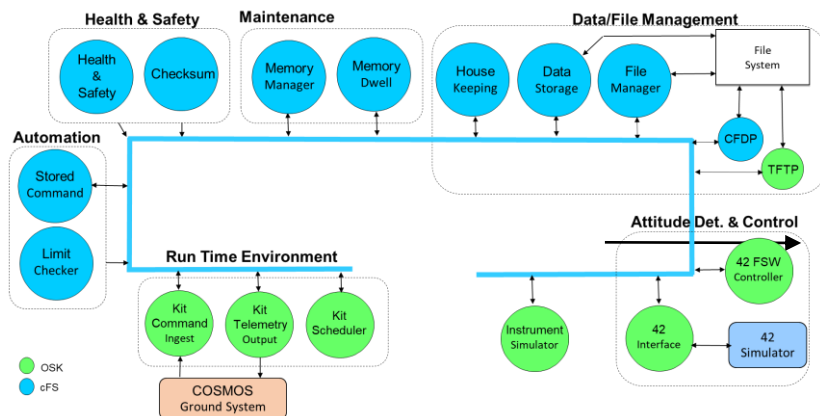
*ConOps  
Req, Hardware*

Iterate over  
FSW System  
Engineering  
Topics

## Mission Model

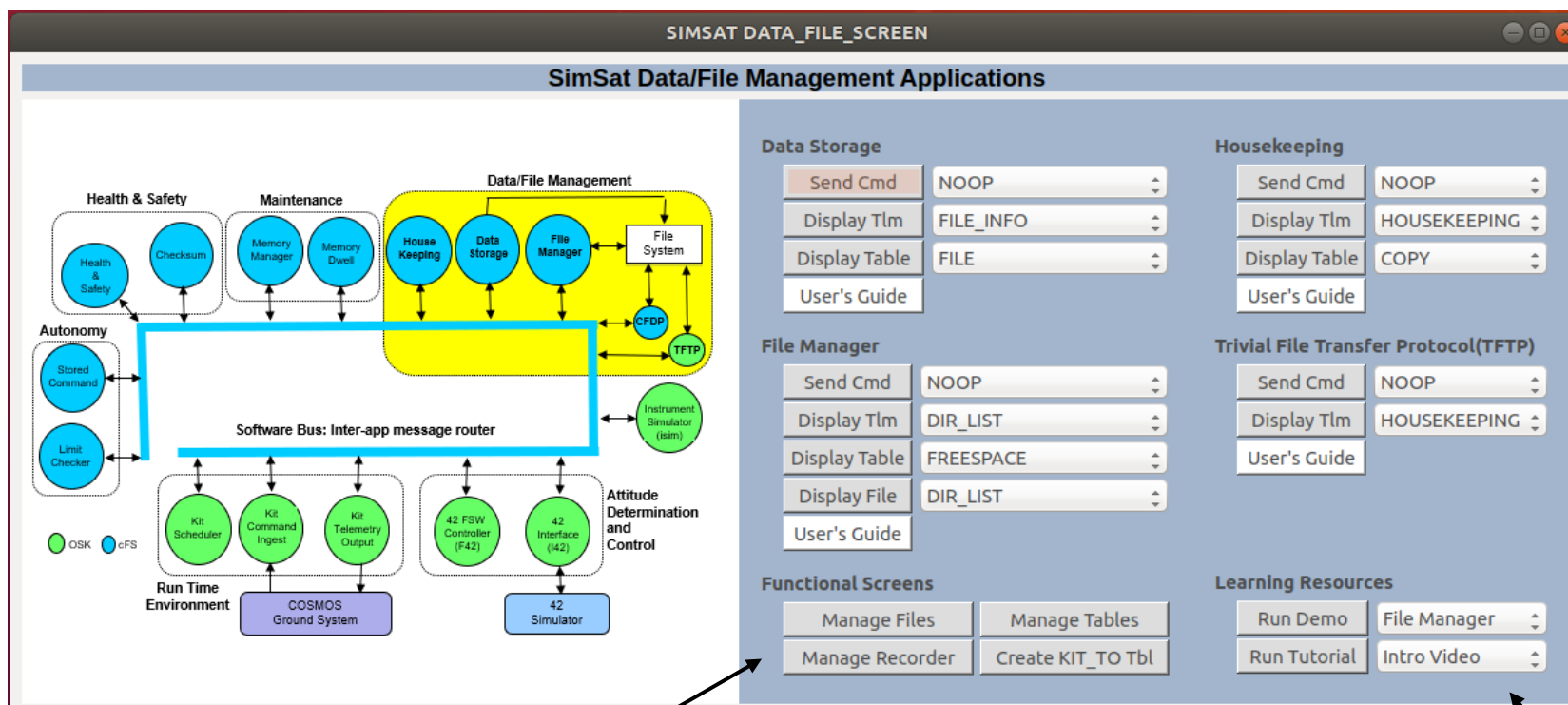


## SimSat Reference Mission



- Workbook style documentation steps users through FSW system engineering (SE) topics
- Applications addressed in small groups that collaborate to provide end-user functionality

Each functional application group screen uses the following layout



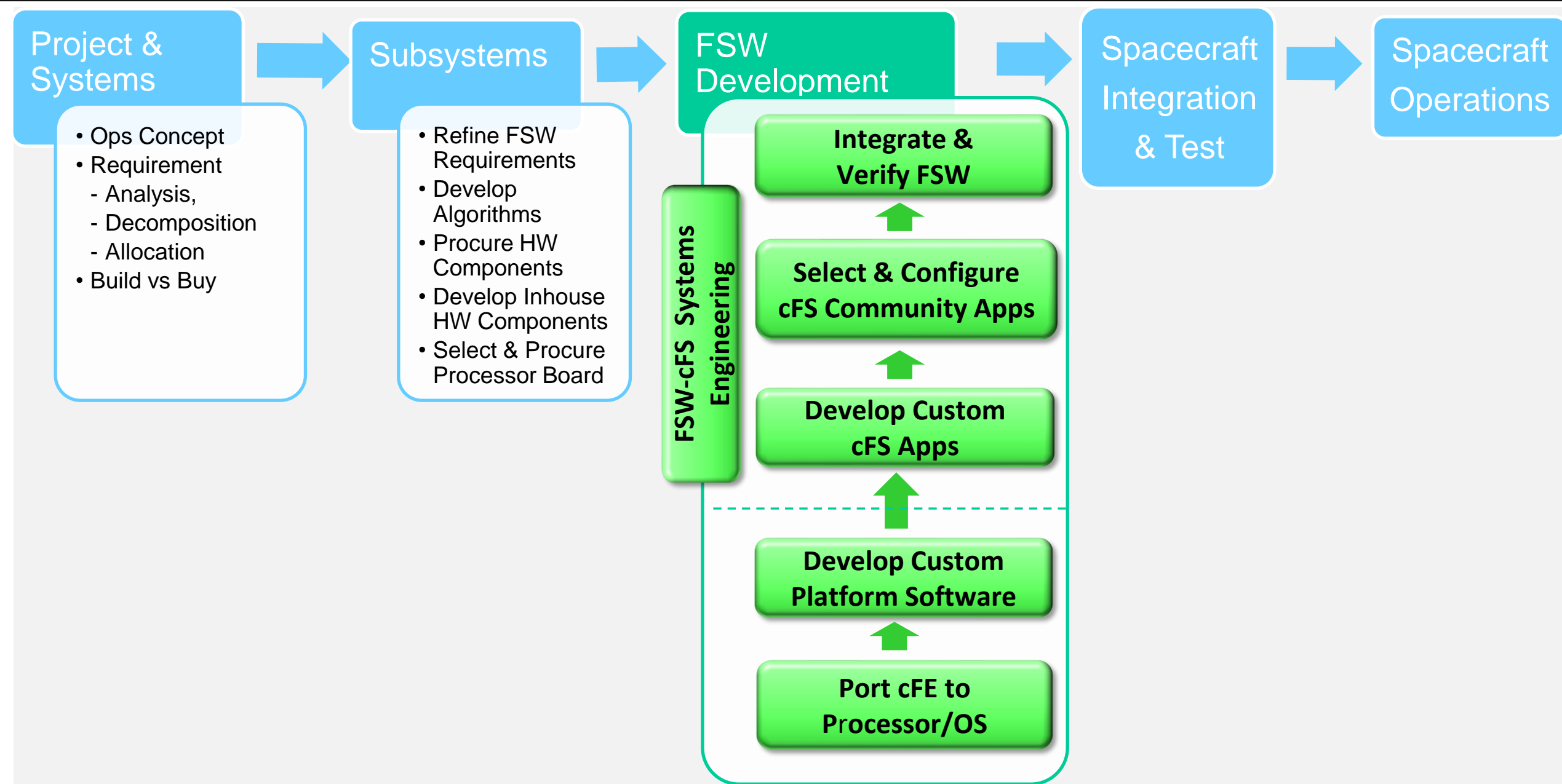
## Complete interface to each app

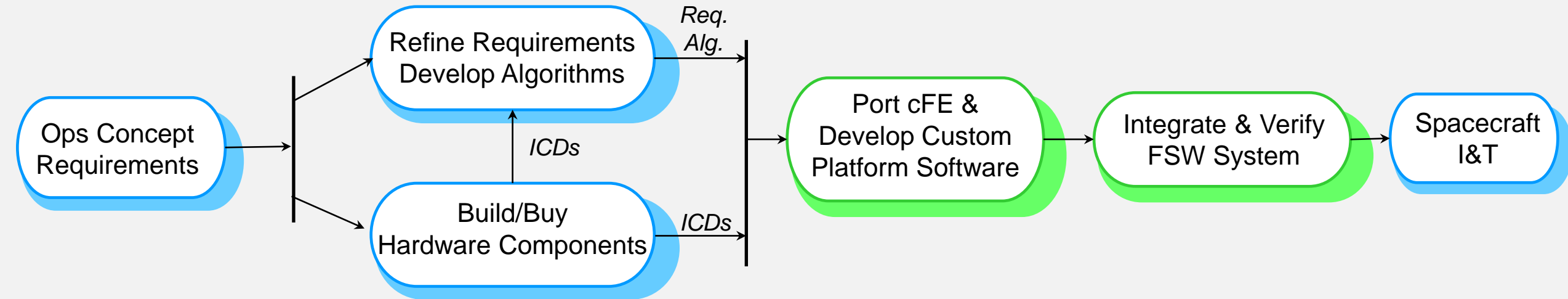
- All commands
- All telemetry packets
- “Display Table” – Dump, transfer and display table in COSMOS Table Manager
- “Display File” – Issue app’s command to create a file, then transfer and display binary file in COSMOS Table Manager

Functional screens combine commands and telemetry from one or more apps that work together to perform a related tasks.

Launch videos, demos (pre-defined screen sequences) and tutorials (slides and/or scripts)

# Use Case #2 – Develop a cFS-based FSW System





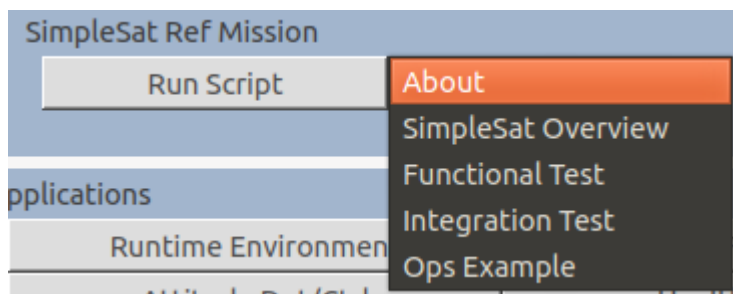
- **OSK does not currently support platform development activities**
- **Preliminary plans to create a Code-As-You-Go (CAYG) cFE porting tutorials**
  - YouTube video with git repos



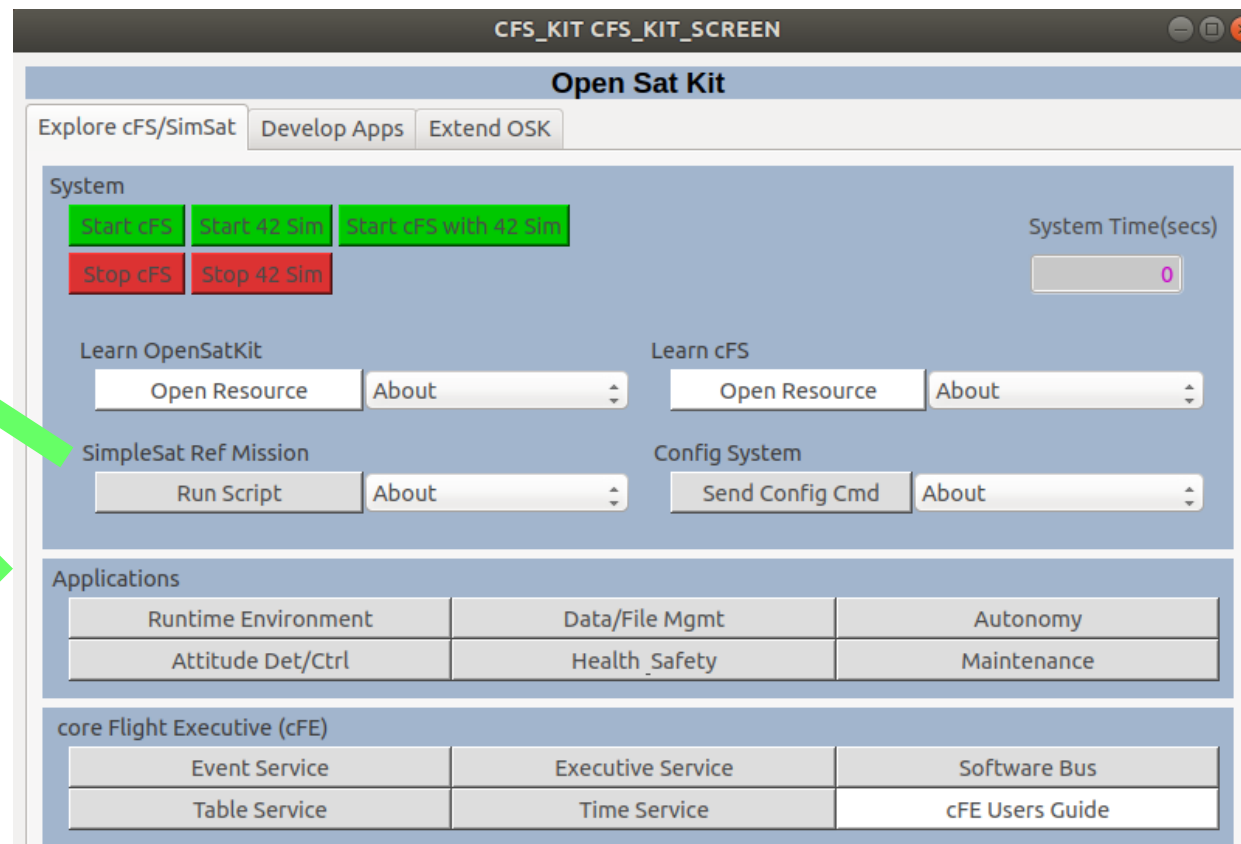
## “Explore cFS/SimSat” Tab

1. SimSat reference mission slides and scripts
2. Launch application functional group screen (next slide)

### 1 SimSat Reference Mission

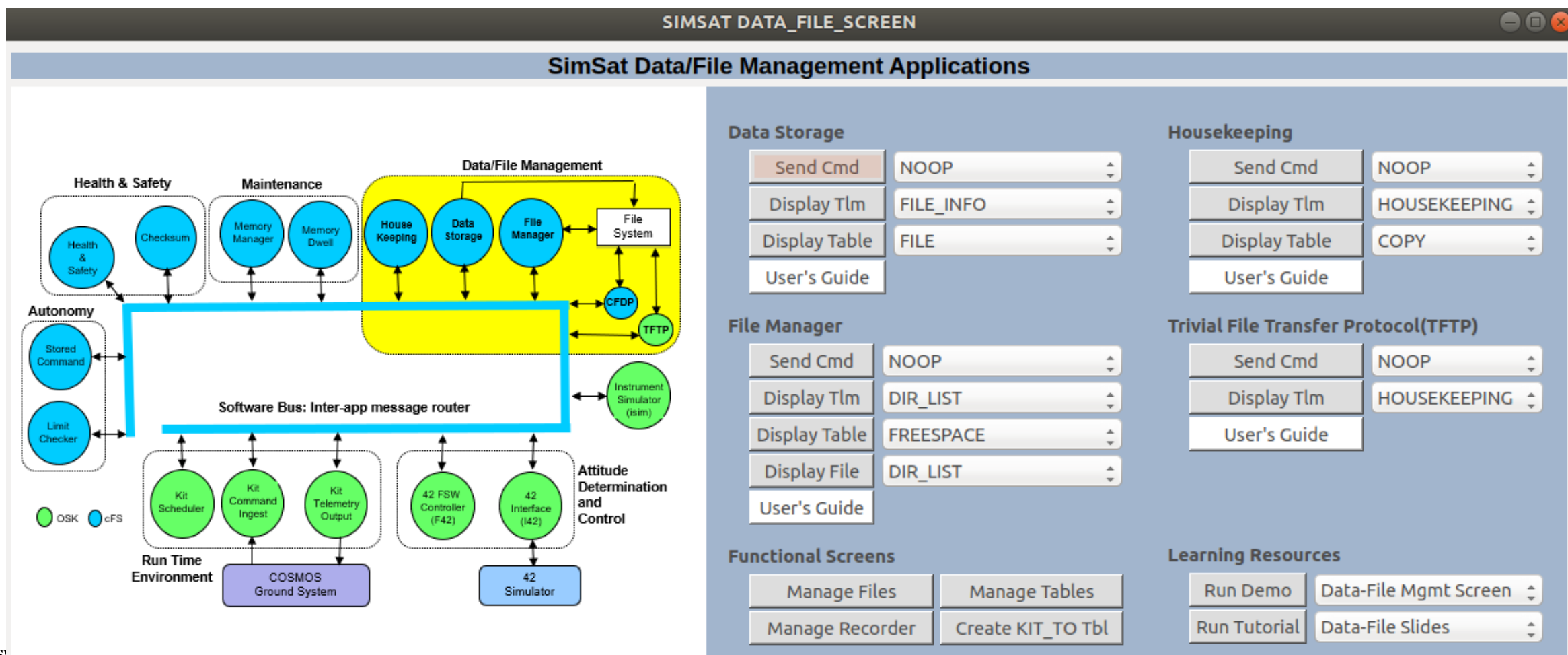


### 2 Application Group (next slide)



- **Objectives**
  - SimSat is a fictional spacecraft that provides a reference mission context
  - Provide a complete application suite illustrating
    - What apps are required to meet a mission's requirements
    - How they are configured and integrated as a system (not all apps configured yet)
    - Current app suite includes command & data handling (C&DH) apps
  - Provide example scripts
    - Integration test script
    - Operational script
  - Provide context for training exercises
- **SimSat is not integrated with the 42-simulator**
  - Closed-loop 42-Simulator example scenario is not related to SimSat

- Each screen highlights the pre-configured SimSat apps that participate in achieving a particular user goal
- Similar organization to the cFE service screens: App ground interface, Functional screen link and a Learn section
- Some demos exist, but not all apps configured for SimSat and no tutorials have been created





# File-Data Storage Demo



OSK-Dev16 Ubuntu 18.04 - VMware Workstation

File Edit View VM Tabs Help

Home OSK-Dev16 Ubuntu 18.04

Activities Xfce Terminal

Sun 14:29

core Flight System

```
on
EVS Port1 42/1/ISIM 100: ISIM App Initialized. Version 1.1.0
EVS Port1 42/1/CS 142: CS Apps Table verification results: good = 0, bad = 0, un
used = 24
EVS Port1 42/1/CS 108: CS Apps Table: No valid entries in the table
EVS Port1 42/1/SC 21: RTS table file load count = 8
EVS Port1 42/1/SC 9: SC Initialized. Version 2.5.0.0
EVS Port1 42/1/CF 1: CF Initialized. Version 2.2.1.0
EVS Port1 42/1/CS 139: CS Tables Table verification results: good = 0, bad = 0,
unused = 24
EVS Port1 42/1/CS 109: CS Tables Table: No valid entries in the table
EVS Port1 42/1/CS 127: OS Text Segment disabled due to platform
EVS Port1 42/1/CS 1: CS Initialized. Version 2.4.0.0
1980-012-14:03:20.59879 ES Startup: CFE_ES_Main entering APPS_INIT state
1980-012-14:03:20.59882 ES Startup: CFE_ES_Main entering OPERATIONAL state
EVS Port1 42/1/CF 73: SemGetId Err:Chan 0 downlink PDUs cannot be throttled.0xFF
FFFFEF
EVS Port1 42/1/CFE_TIME 21: Stop FLYWHEEL
EVS Port1 42/1/KIT_TO 306: Telemetry output enabled for IP 127.0.0.1
EVS Port1 42/1/SC 73: RTS Number 001 Started
EVS Port1 42/1/SC 86: RTS 001 Execution Completed
EVS Port1 42/1/KIT_SCH 404: Major Frame Sync too noisy (Slot 1). Disabling synch
ronization.
g
```

CFS\_KIT CFS\_KIT\_SCREEN

Open Sat Kit

Explore cFS/SimSat Develop Apps Extend OSK

System

Start cFS Start 42 Sim Start cFS with 42 Sim

Stop cFS Stop 42 Sim

System Time(secs)

1001038

Learn OpenSatKit

Open Resource About

Learn cFS

Open Resource About

SimpleSat Ref Mission

Run Script About

Config System

Send Config Cmd About

Applications

Runtime Environment	Data/File Mgmt	Autonomy
Attitude Det/Ctrl	Health Safety	Maintenance

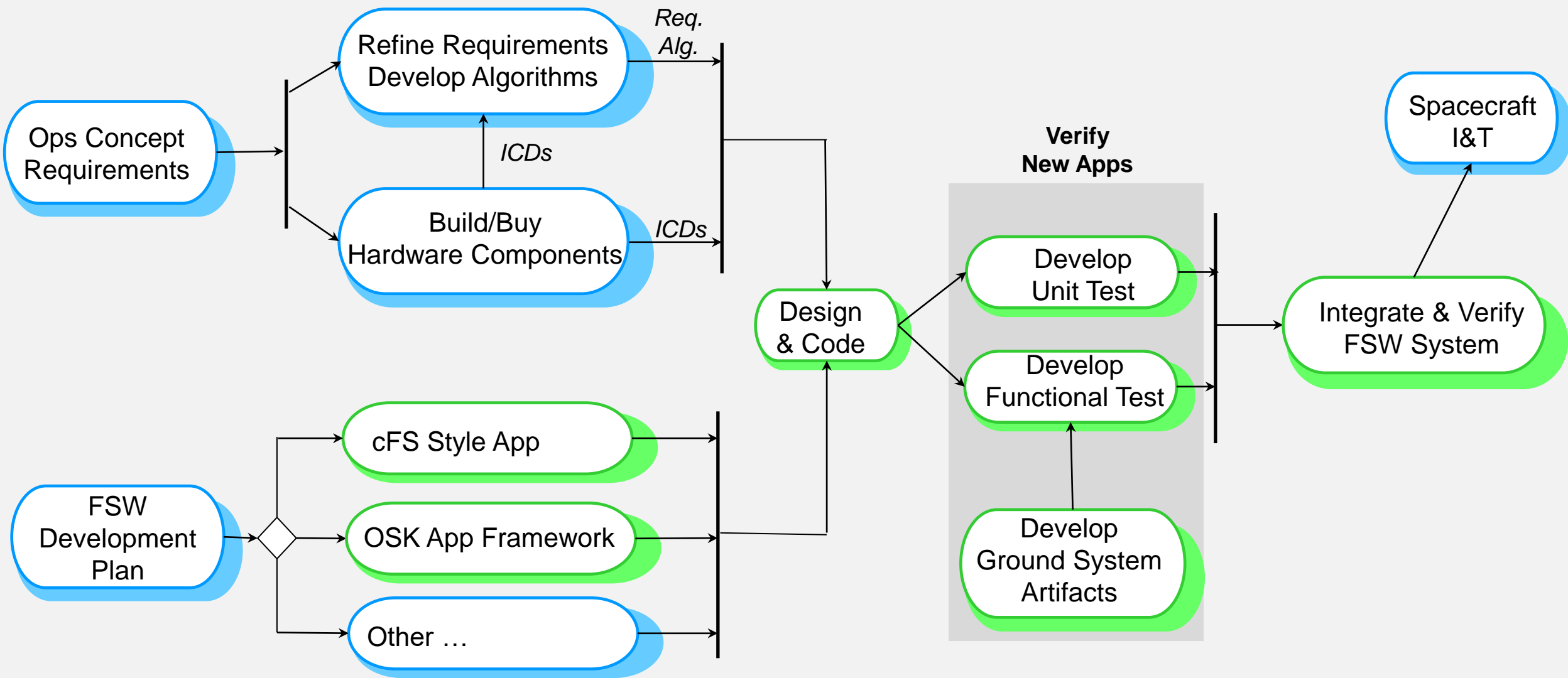
core Flight Executive (cFE)

Event Service	Executive Service	Software Bus
Table Service	Time Service	cFE Users Guide

Flight Event Messages

Major Frame Sync too noisy (Slot 1). Disabling synchronization.







# Developing New Apps



- **OSK app generator creates “hello world” app source code and COSMOS command & telemetry definition files with templates for**
  1. cFS User’s Guide Style App
    - Design and code according to the cFS Developer Guide
  2. OSK Framework App
    - Adhere to cFS API principles but design according to the OSK framework design patterns and coding idioms
- **Initial app development often done independent of COSMOS**
  - “Hello world” tool useful for learning, but often useful to start with an existing app that has a top-level design close to your needs
  - 42 Interface (I42) and 42 FSW Controller (F42) serve as high level example
- **Verify New Apps**
  - OSK does not augment cFS’ unit testing framework
  - OSK supports functional app testing and system integration testing using COSMOS Script Runner and Test Runner tools
    - Thin integration and functional test infrastructure built on top of COSMOS script API
    - Minimal example tests, more planned
  - Ground tools written as needed



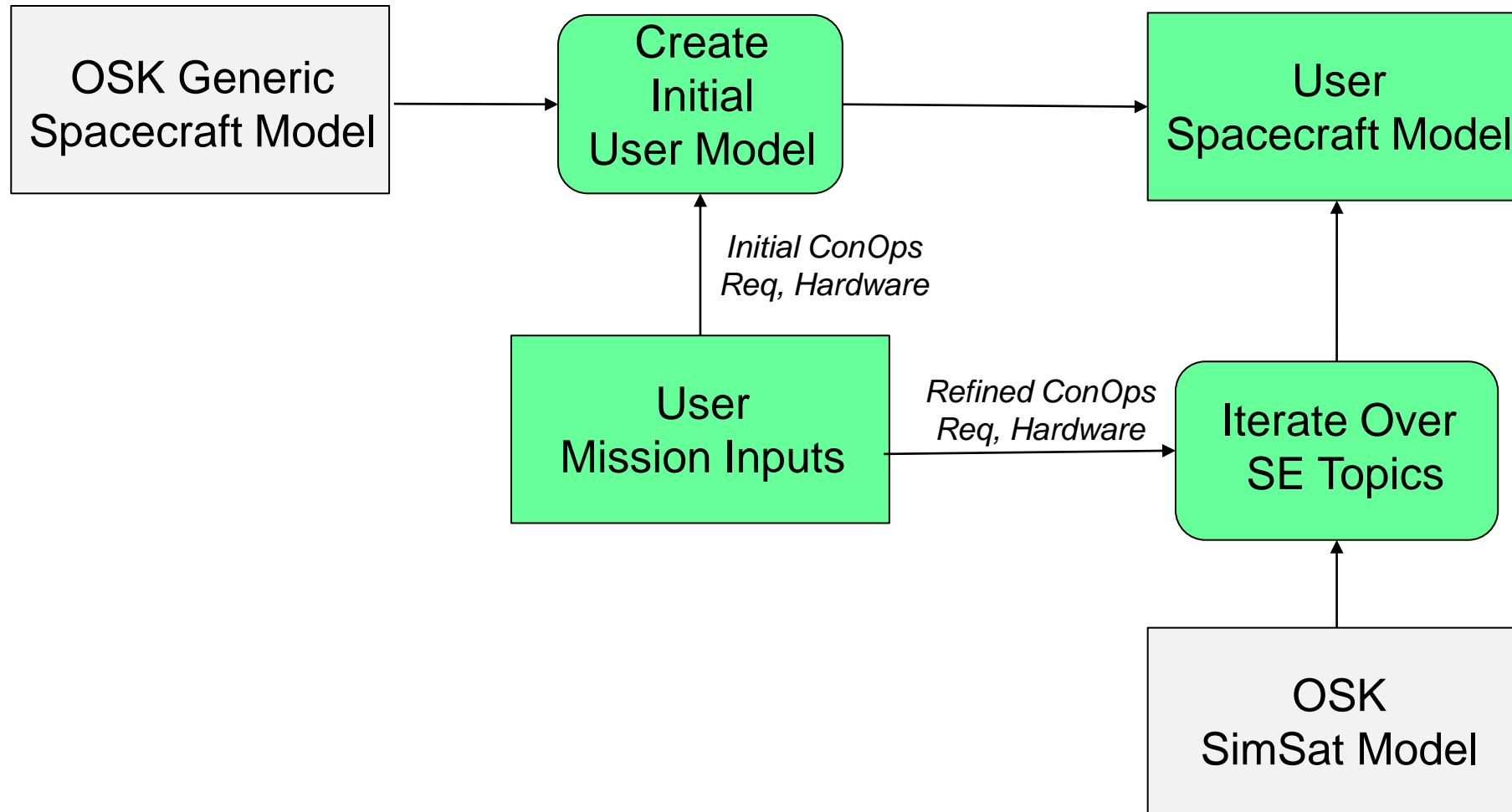
# Introduction (2 of 2)

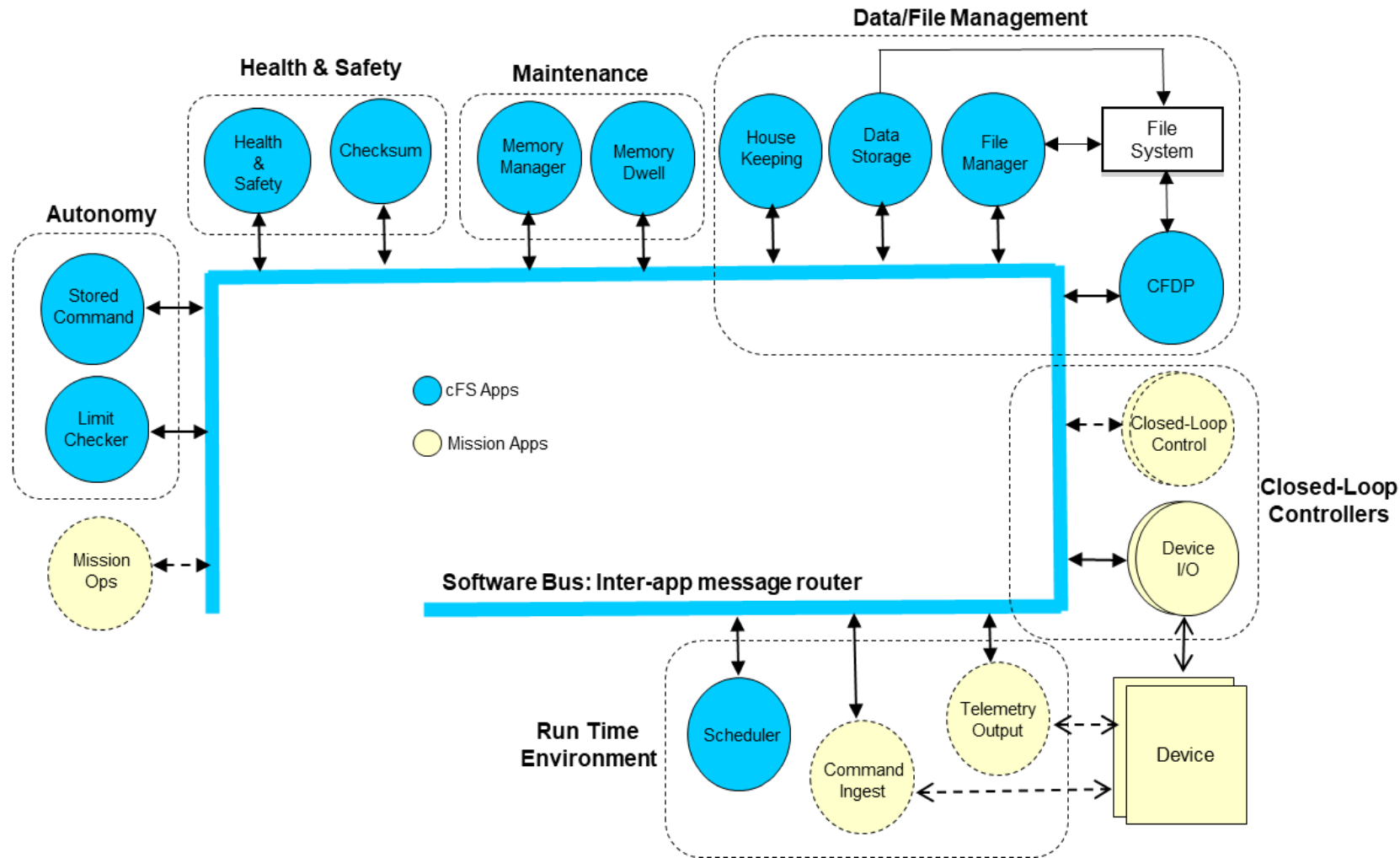


- OSK supports integrating and testing cFS community apps, OSK apps, and user mission-specific apps into a functional FSW system that runs within OSK's software-in-the-loop (SIL) environment
- Nothing precludes OSK from being used in later mission lifecycle phases, however, creating the hardware in-the-loop (HIL) interfaces, developing simulators, and migrating ground system artifacts (if COSMOS is not used) are not covered by OSK.
- These efforts are represented by the gray arrows. The green arrow pointing to the processor card is not within the OSK boundaries because porting the cFE to a hardware platform is not directly covered by OSK, however, a cFS community platform list <https://github.com/OpenSatKit/cfs-platform-list> is maintained by the OSK project and provides links to cFS porting resources.
- OSK is not required to develop cFS apps, however, note the following
  - You will eventually need a ground interface and test script environment
  - You can leave OSK's SimSat environment and develop new apps in a new mission or target or use OSK's Sandbox target

- 1. Create an initial app model from OSK's generic spacecraft model using mission concepts of operations, mission requirements, and the spacecraft hardware architecture often in the form of a block diagram**
  - Initial goal is to create a "good enough" architecture based on the maturity of the information at hand
  - Designing FSW is a very iterative process with top-down and bottom-up technical and non-technical forces at work
  - Trades are often made throughout the requirements analysis and spacecraft design phases that impact FSW. These forces are both technical and non-technical concerns may also influence decisions that impact the technical design
- 2. Analyze OSK's SimSat mission app model to understand what capabilities exist within OSK**
- 3. Work through system engineering topics and app groups to design new apps and understand how to configure cFS community apps**
- 4. Work through spacecraft lifecycle to determine the need for**
  - Different versions of apps for different test environments
  - Simulation apps that can serve
- 5. Determine how you want to use OSK in the spacecraft lifecycle**
  - Create OSK mission target
  - Plan migration to PIL and other environments
- 6. If OSK will be used in a verification role then develop test artifacts as needed**
  - FSW validation should occur in a high-fidelity test environment









# OSK Generic Spacecraft Model



- **Show how generic model can be tailored for different missions. First focus on CubeSats, larger mission can extrapolate.**
- **Three main tailoring areas**
  - Device I/O
  - Closed loop control needs
  - Mission ops
- **Examples**
  - COTS vs inhouse ADCS
  - Payload control (closed vs open loop) and data management
  - Need for a mission manager app or ACS mode manager type app coupled with autonomy app group to achieve con ops
- **In order to work through the steps an example user mission is needed to show how to create a user model and then create a plan for how to migrate from SimSat to the user model needs**



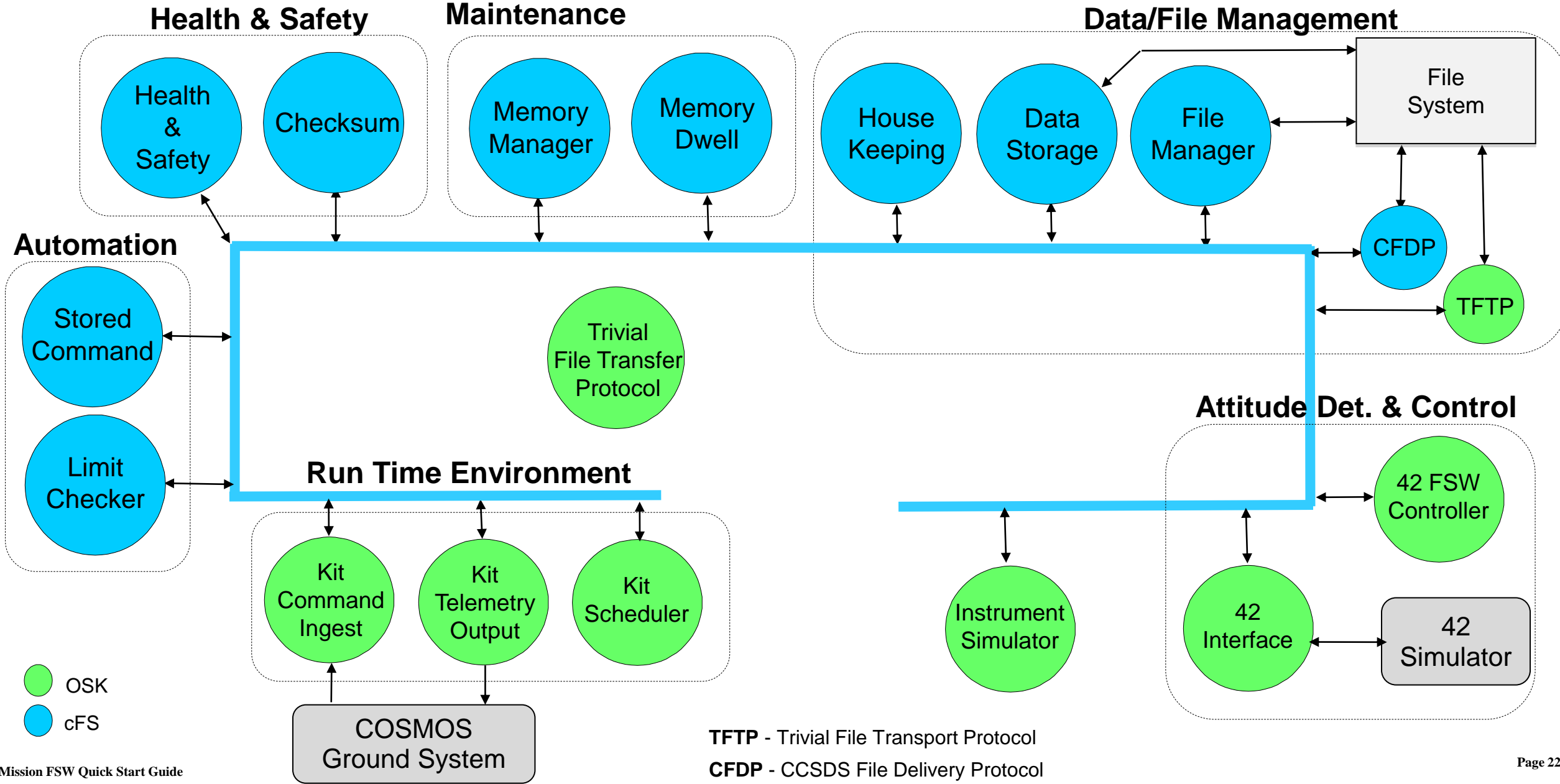
# System Engineering Topics



- **System Engineering topics do not have a one-to-one correlation with app groups**
- **ConOps, mission requirements, and a hardware block diagram are required inputs**
  - This is an iterative process so reapply these inputs
  - Spacecraft lifecycle can be considered a process input and iteratively examined to determine what's needed
- **Goal is to manage complexity by working through topics**
  - Topics are not 100% orthogonal (non-overlapping)
  - Create a ConOps and show how it impacts different apps groups
- **SE Topics**
  1. Device I/O
  2. Close-loop control
  3. Mission Ops & autonomy
  4. Data-File management
  5. Runtime Environment
  6. FDIR
  7. Interface & control apps
  8. Maintenance
  9. Time
  10. Parameters and configuration
- **V&V Topics**
  1. Unit tests
  2. App functional tests
  3. Integration tests
  4. System and ops tests



- **Default OSK app configuration is for a fictitious satellite called SimpleSat (SimSat)**
  - The cFS can be used for many different types of embedded systems. A spacecraft was chosen due to the increased usage of the cFS on CubeSats
- **SimSat provides a reference mission to provide context to**
  - Illustrate what applications are required and how they are configured and integrated as a system to meet the requirements
  - Demonstrate an example integration test script
  - Demonstrate an operational script
- **This does not include**
  - Porting SimSat to a new platform
  - Integrating hardware devices
- **SimSat is a**
  - Low Earth Orbit (LEO) satellite with one nadir-pointing science instrument
  - The instrument has
    - A detector that produces 10 bytes of data per second
    - A power the following sequence: Apply power, wait for instrument initialization (~20s), and command to enable science
  - The science team requires



- **The previous slide shows a cFS “bubble” chart where each app is a bubble and they communicate via messages on the software bus.**
  - The blue cFS apps are reusable open source apps that are available on <https://github.com/nasa/xx> where ‘xx’ is the abbreviated app name
  - The green OSK apps were written specifically for OSK
  - The external COSMOS and 42 interfaces use UDP and TCP respectively
- **Apps are designed to perform a dedicated function with clear interfaces and they operate in groups to achieve higher level mission objectives**
- **Runtime Environment Apps**
  - **Kit Command Ingest (KIT\_CI)** receives CCSDS command packets from COSMOS and sends them on the Software Bus
  - **Kit Telemetry Output (KIT\_TO)** reads CCSDS telemetry packets from the Software Bus and sends them to COSMOS
  - **Kit Scheduler (KIT\_SCH)** contains tables that define when to send messages on the Software Bus
    - Apps can use these messages to perform synchronous activities, e.g. sending their housekeeping status packet

- **Data/File Management**

- **File Manager (FM)** provides a ground interface for performing common directory and file operations
- **Data Storage (DS)** reads packets from the software bus and writes them to files according to table-defined
- **Housekeeping (HK)** creates new telemetry packets from pieces of other telemetry packets. The new packets are written to the SB and can be stored and/or telemetered.
- **Trivial File Transfer Protocol (TFTP)** transfers files between the flight and ground COSMOS. There's an open source CCSDS File Delivery Protocol (CFDP) app that will be added in a future release.

- **Autonomy**

- **Limit Checker (LC)** monitors one or more telemetry values and start stored command relative time sequences (RTSs) in response to limit violations
- **Stored Command (SC)** Provides services to execute preloaded, table-defined command sequences at predetermined absolute or relative time intervals





- **Attitude Determination and Control Apps**

- **42 Interface (I42)** manages a TCP/IP connection to 42 and transfers actuators/sensor packets to/from 42
- **42 FSW (F42)** Implements the “ThreeAxisFsw” attitude control algorithm defined in 42

- **Maintenance**

- **Memory Dwell (MD)** creates telemetry packets containing contents of memory location specified in dwell tables
- **Memory Manager (MM)** provides read/write access to memory

- **Health & Safety**

- **Checksum (CS)** monitors checksums across table-defined static code/data regions and reports errors
- **Health & Safety (HS)** monitors table-defined application check-in and event messages and reporting errors and/or starting a RTS to address the issue



# Apply the System Engineering Development Processes





# System Engineering Topics



- **SE Topics**
  1. Device I/O
  2. Close-loop control
  3. Mission Ops & autonomy
  4. Data-File management
  5. Runtime Environment
  6. FDIR
  7. Interface & control apps
  8. Maintenance
  9. Time
  10. Parameters and configuration
- **V&V Topics**
  1. Unit tests
  2. App functional tests
  3. Integration tests
  4. System and ops tests

- 1. What data must be downlinked to meet mission goals?**
- 2. What are the contact frequencies, durations, and data rates?**
- 3. What parts of operations need to be automated?**
- 4. What level of fault detection, isolation, and recovery (FDIR) is necessary?**
  - Bottom-up**
    - 1. What processor card is being used and is a realtime operating system required?**
    - 2. What device interfaces does the FSW need to manage and how are they connected?**
    - 3. Make versus buy decisions. Some top-down decisions impact bottom-up design.**





CFS\_KIT CFS\_KIT\_SCREEN

OpenSatKit

Mission FSW

cFS FSW Edu

PiSat

R&D

System - simsat Target

Start cFS

Start 42 Sim

Start cFS with 42 Sim

Stop cFS

Stop 42 Sim

System Time(secs)

1011515

Config System

Send Config Cmd

About

OSK Docs & Videos

Open Resource

About

Docs & Videos

Open Resource

About

Application Groups

Runtime Environment	Data/File Mgmt	Autonomy
Attitude Det/Ctrl	Health_Safety	Maintenance

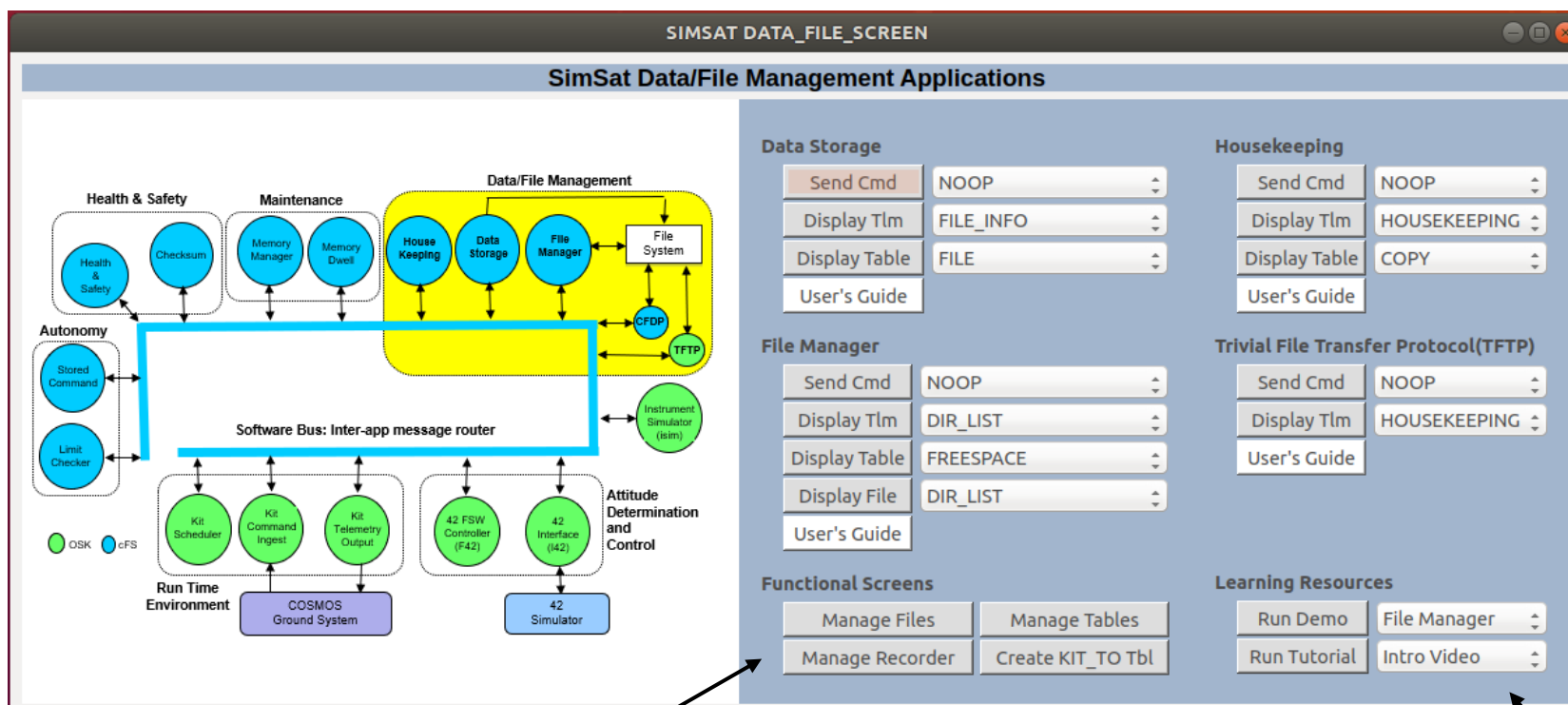
Tune, Verify, and Validate

Perf Mon Mgmt	Perf Mon Demo	Run Unit Tests	Run Intgr Test
Run App Func Tests	Manually Select	Run Ex Sys/Ops Scripts	Two Ground Passes

Flight Event Messages



Each functional application group screen uses the following layout



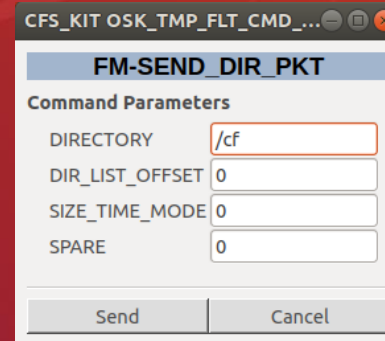
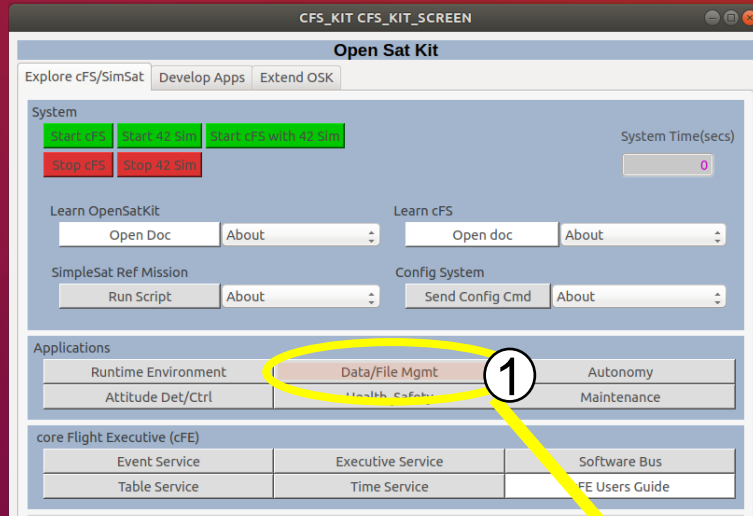
## Complete interface to each app

- All commands
- All telemetry packets
- “Display Table” – Dump, transfer and display table in COSMOS Table Manager
- “Display File” – Issue app’s command to create a file, then transfer and display binary file in COSMOS Table Manager

Functional screens combine commands and telemetry from one or more apps that work together to perform a related tasks.

Launch videos, demos (pre-defined screen sequences) and tutorials (slides and/or scripts)

1. Launch Data/File Management Screen from OSK main screen
2. Access FM commands, telemetry, tables, files and users guide.



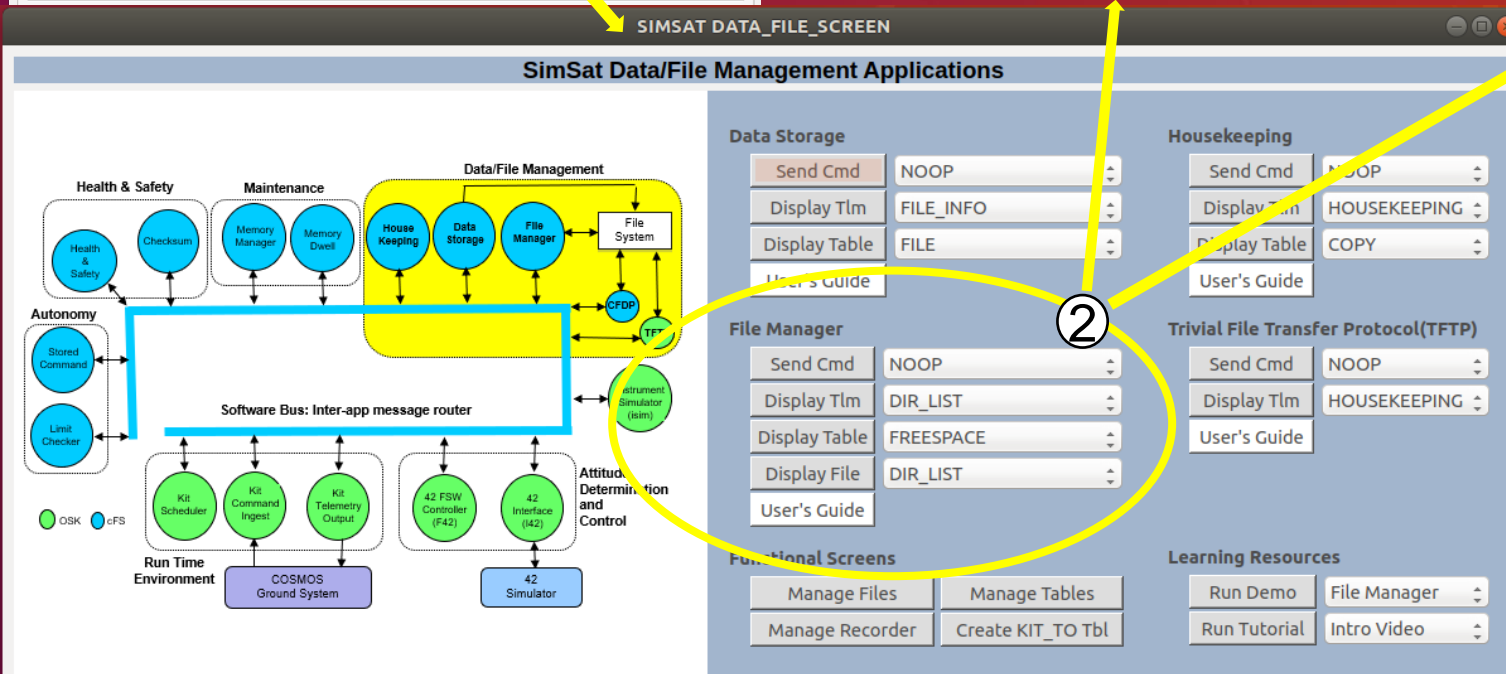
Packet Viewer : Formatted Telemetry with Units

File View Help

Target: FM Packet: DIR\_LIST\_PKT

Description: Get Directory Listing telem

	Item	Value
1	*PACKET_TIMESECONDS:	1606580416.641317
2	*PACKET_TIMEFORMATTED:	2020/11/28 08:20:16.641
3	*RECEIVED_TIMESECONDS:	1606580416.641317
4	*RECEIVED_TIMEFORMATTED:	2020/11/28 08:20:16.641
5	*RECEIVED_COUNT:	1
6	CCSDS_STREAMID:	0x088C
7	CCSDS_SEQUENCE:	49153
8	CCSDS_LENGTH:	1601
9	CCSDS_SECONDS:	1001211
10	CCSDS_SUBSECS:	65115
11	DIRNAME:	/cf
12	TOTALFILES:	94
13	PACKETFILES:	20
14	FIRSTFILE:	0
15	FILE0_NAME:	bm.so
16	FILE0_SIZE:	0
17	FILE0_MOD_TIME:	0
18	FILE0_MODE:	0
19	FILE1_NAME:	cf.so



**SimSat Data/File Management Applications**

**Data Storage**

Send Cmd	NOOP
Display Tlm	FILE_INFO
Display Table	FILE
User's Guide	

**Housekeeping**

Send Cmd	NOOP
Display Tlm	HOUSEKEEPING
Display Table	COPY
User's Guide	

**File Manager**

Send Cmd	NOOP
Display Tlm	DIR_LIST
Display Table	FREESPACE
Display File	DIR_LIST
User's Guide	

**Trivial File Transfer Protocol(TFTP)**

Send Cmd	NOOP
Display Tlm	HOUSEKEEPING
User's Guide	

**Functional Screens**

Manage Files	Manage Tables
Manage Recorder	Create KIT_TO Tbl

**Learning Resources**

Run Demo	File Manager
Run Tutorial	Intro Video

**CFS\_KIT FILE\_MGMT\_SCREEN**

**File Management**

**Directory Management**

Create	Delete
List to Packet	Write to File

**File Manager Directory Listing**

DIRNAME:	
TOTALFILES:	0
PACKETFILES:	0
FIRSTFILE:	0
FILE0_NAME:	
FILE1_NAME:	
FILE2_NAME:	
FILE3_NAME:	
FILE4_NAME:	
FILE5_NAME:	
FILE6_NAME:	
FILE7_NAME:	
FILE8_NAME:	
FILE10_NAME:	
FILE11_NAME:	

**File Management**

Copy	Move
Rename	Decompress
Delete	Delete All
Concat	Get Info
List Open	

**File Manager Housekeeping**

Cmd Valid Cnt	0
Cmd Error Cnt	0
Child Cmd Valid Cnt	0
Child Cmd Error Cnt	0

1. FM summary page with HK and directory listing

2. FM feature demo

3. YouTube Tutorials

**Community Apps** ▶ PLAY ALL

Tutorials for configuring and using cFS community apps to meet your needs

**OSK Data & File Management Apps Intro** 10:05

**OSK Runtime Environment Apps** 22:43

**CFS\_KIT FILE\_MGMT\_DEMO\_SCREEN**

**File Management Demo**

This demo shows some basic file management features. It uses the Trivial File Transport Protocol (TFTP) App to transfer files between COSMOS and the cFS. It uses the File Manager (FM) App to manage flight directories and files. Click...

<More Info> to obtain more information about the current step  
<Demo> to issue commands to demonstrate a feature in the current step  
<Next> to move to the next step

More Info Demo Next ->

**Script Runner** : /mnt/hgfs/OpenSatKit/cosmos/config/targets/FM/procedures/demo\_fm\_features.rb

demo\_fm\_features.rb

Stopped

Start Pause Stop

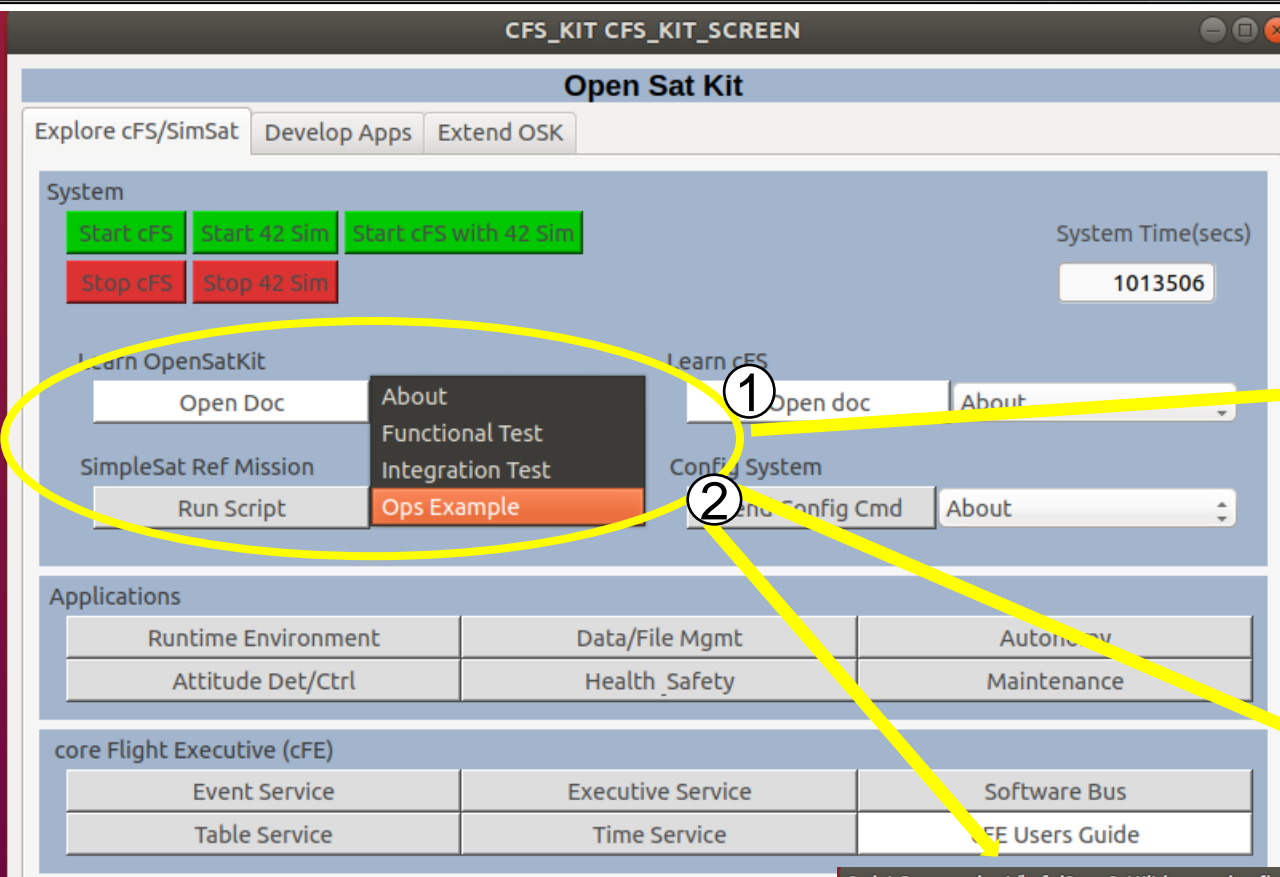
```

1 #####
2 # File Manager Feature Demo
3 #
4 # Notes:
5 # 1. Developed for the YouTube File Manager training video and
6 # originally based the CFS_KIT File Manager demo
7 # 2. Debug events are enabled for the apps used during the demo.
8 # COSMOS cmd() is used instead of OSK App.send_cmd() because speed is
9 # preferred over command verification
10 #

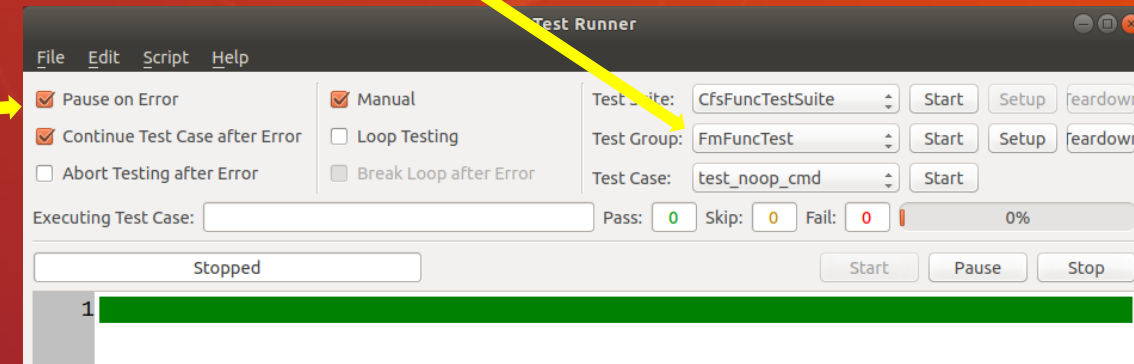
```

Script Output:

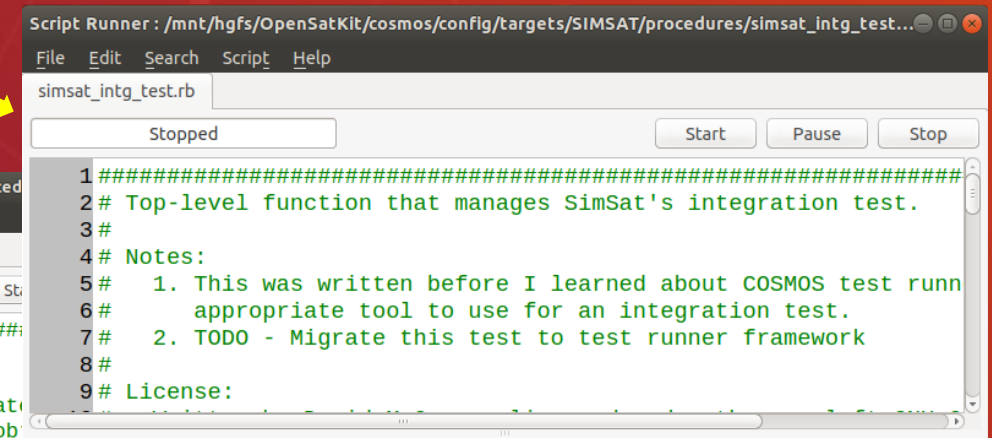
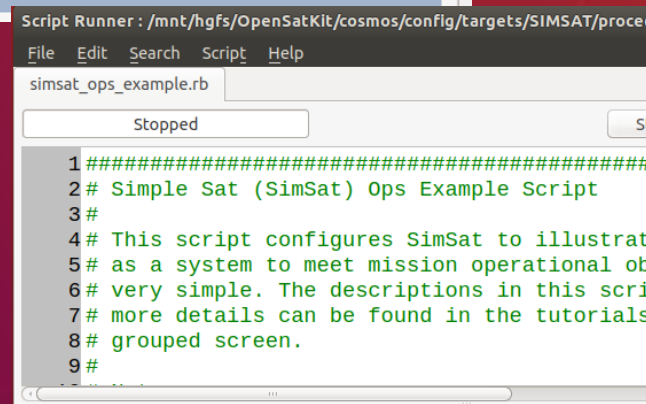




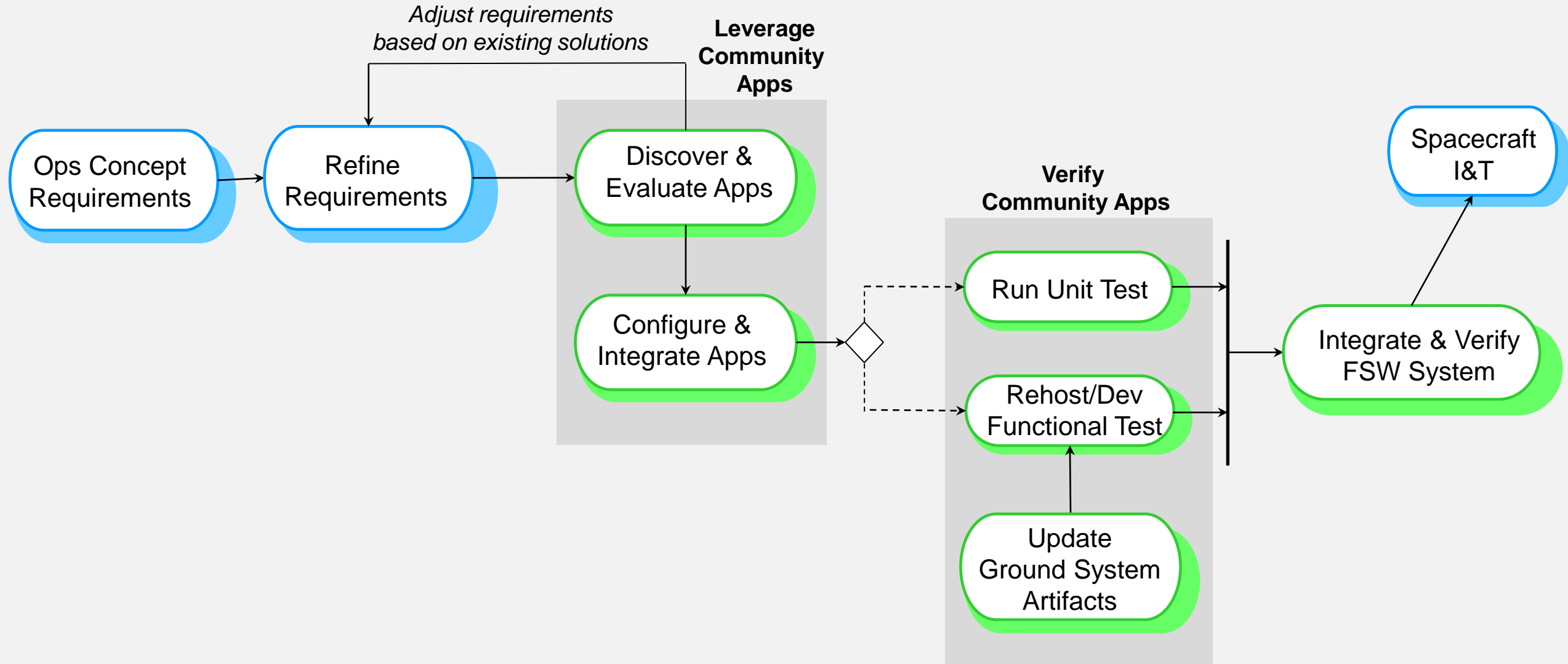
1. Functional Test Suites contain Test Groups for each app that run in the COSMOS Test Runner



2. The Integration Script and Ops Example use the COSMOS Script Runner



# Select and Configure cFS Community Apps





- **Leverage Community Apps**
  - OSK's SimSat application groups provide a system context for users to learn how apps work together to achieve a goal
    - Slides, YouTube videos, and built-in demos
    - Example operational script
    - The above resources are not complete, and the plan is to continually update them
  - Prototype tools for generating default Data Storage, KIT\_TO, and Housekeeping table source files
    - Links to tools from application group screens
  
- **Verify Community Apps**
  - NASA app unit tests can be run as needed
  - OSK apps do not currently have unit tests
  - OSK supports functional app testing and system integration testing using COSMOS Script Runner and Test Runner tools
    - Thin integration and functional test infrastructure built on top of COSMOS script API
    - No functional tests exist



# OSK SimSat NASA cFS Libs & Apps



Name	Description	Cmd, Tlm, Tbl Defined	Video	Notes
<b>CFDP (CF)</b>	Implements the CCSDS File Delivery Protocol (CFDP)	Partial	No	Compiled, but not loaded by default. Downlink flow control semaphore not implemented
<b>CFS_LIB</b>	Utility functions	N/A	No	
<b>Checksum (CS)</b>	Compute checksums across table-defined memory blocks	All	No	
<b>Data Storage (DS)</b>	Read packets from the software bus and store them in files	All	Group	
<b>File Manager (FM)</b>	Provide basic directory & file management services	All	Group Solo	
<b>Housekeeping (HK)</b>	Combine parts of packets into new packets	All	Group Solo	
<b>Health &amp; Safety (HS)</b>	Provide system health & safety checks	All	No	
<b>Limit Checker (LC)</b>	Monitor values in packets and activate stored commands as needed	All	No	
<b>Memory Dwell (MD)</b>	Create telemetry packets containing user defined memory locations	All	No	
<b>Memory Manager (MM)</b>	Provide interface to read/write (load/dump) memory locations.	All	No	Use with caution!
<b>Software Bus Network (SBN)</b>	Software Bus Network	None	No	Compiled, but not configured or loaded by default.
<b>Stored Command (SC)</b>	Manage absolute time and relative time command sequences		No	



# OSK SimSat Custom Libs & Apps



Name	Description	Cmd, Tlm, Tbl	Video	Notes
<b>expat_lib</b>	XML Parser	n/a	No	Only used by demo apps
<b>osk_c_fw</b>	Application framework for apps written in C	n/a	No	Used for all OSK SimSat apps
<b>osk_42_lib</b>	Utilities to manage 42's interface	n/a	No	Allows default options to be used for 42's socket interface code generation utility & 42's acapp to be used unchanged
<b>F42</b>	Manages execution of 42's "acapp" controller	All	No	
<b>I42</b>	Manages sensor/actuator interfaces for F42	All	No	
<b>ISIM</b>	Simulates an instrument for SimSat demos	All	No	
<b>KIT_CI</b>	Ingest commands over UDP socket and send them on software bus	All	Group	Functionally similar NASA's lab_ci
<b>KIT_SCH</b>	Send table-defined messages on the software bus	All	Group	Schedule algorithm similar to sch_lab. Message and scheduler table implemented in JSON.
<b>KIT_TO</b>	Receive telemetry packets from the software bus and output them over a UDP socket	All	Group	Output JSON filter table determines packet output rates. OSK table tool can generate the table for a CSV file.
<b>TFTP</b>	Implement Trivial File Transport Protocol	All	No	Default file transfer protocol. Doesn't preclude use of CF app.



# OSK cFS Additional Libs & Apps



Name	Description	Cmd, Tlm, Tbl Defined	Video	Notes
<b>osk_cpp_lib</b>	Application framework for apps written in C++	n/a	No	Prototype only used by C++ demo app
<b>mqtt_lib</b>	MQTT utilities	n/a	No	Unchanged from Alan Cudmore's repo: <a href="https://github.com/alanc98/mqtt_lib">https://github.com/alanc98/mqtt_lib</a>
<b>Benchmark (BM)</b>	Provides Dhrystone, Whetstone, and custom cFS measurements	All	No	Prototype needs review and documentation.
<b>File Manager (FileMgr)</b>	Same functions as NASA's FM	All	No	Refactor of NASA's FM app using osk_c_fw. Investigates moving most config parameters to a runtime startup file.
<b>Heater Control (HC)</b>	Simple bang-bang controller for demo purposes	All	No	Used in Limit Checker demo
<b>Heater Sim (HSIM)</b>	Simple heater simulation to provide feedback to HC app	All	No	Used in Limit Checker demo
<b>MQTT</b>	Bridge between a MQTT broker and the software bus	All	No	Early prototype with partial MQTT implementation. Refactor of Alan Cudmore's app from repo: <a href="https://github.com/alanc98/mqtt_app">https://github.com/alanc98/mqtt_app</a> .
<b>osk_c_demo</b>	Sample app to illustrate how to use osk_c_fw	All	No	Not all osk_c_fw features demonstrated. Does not include JSON initialization configuration file.
<b>osk_cpp_demo</b>	Sample app to illustrate how to use osk_cpp_fw	All	No	Preliminary prototype.

- **Exploratory stage of development**
- **Current efforts**
  - Exploring use of runtime rather than compile-time configurations
  - Defining app metadata in JSON file
  - Interactive GUI to manage app discovery and dynamic loading/ unloading
- **Plans**
  - Evaluate CCSDS Electronic Data Sheets
    - Joe Hickey extensions to GRC, <https://github.com/jphickey/edslib>
  - Work with the cFS community in defining app packaging specs



OSK-Dev16 Ubuntu 18.04 - VMware Workstation

File Edit View VM Tabs Help

Home OSK-Dev16 Ubuntu 18.04

Sun 14:06

Activities TlmViewer

Firefox Trash

core Flight System

```
File Edit View Terminal Tabs Help
EVS Port1 42/1/CF 1: CF Initialized. Version 2.2.1.0
EVS Port1 42/1/CS 139: CS Tables Table verification results: good = 0, bad = 0,
unused = 24
EVS Port1 42/1/CS 109: CS Tables Table: No valid entries in the table
EVS Port1 42/1/CS 127: OS Text Segment disabled due to platform
EVS Port1 42/1/CFE_SB 25: Pipe Overflow,MsgId 0x808,pipe KIT_TO_PKT_PIPE,sender
CS
EVS Port1 42/1/CS 1: CS Initialized. Version 2.4.0.0
EVS Port1 42/1/CFE_SB 25: Pipe Overflow,MsgId 0x808,pipe KIT_TO_PKT_PIPE,sender
CS
1980-012-14:03:20.58987 ES Startup: CFE_ES Main entering APPS_INIT state
1980-012-14:03:20.58988 ES Startup: CFE_ES Main entering OPERATIONAL state
EVS Port1 42/1/CF 73: SemGetId Err:Chan 0 downlink PDUs cannot be throttled.0xFF
FFFFEF
EVS Port1 42/1/CFE_TIME 21: Stop FLYWHEEL
EVS Port1 42/1/KIT_SCH 406: Multiple slots processed: slot = 0, count = 2
EVS Port1 42/1/KIT_TO 306: Telemetry output enabled for IP 127.0.0.1
EVS Port1 42/1/SC 73: RTS Number 001 Started
EVS Port1 42/1/SC 86: RTS 001 Execution Completed
EVS Port1 42/1/CFE_TIME 21: Stop FLYWHEEL
EVS Port1 42/1/CFE_TIME 20: Start FLYWHEEL
EVS Port1 42/1/KIT_SCH 404: Major Frame Sync too noisy (Slot 1). Disabling synch
ronization.
```

CFS\_KIT CFS\_KIT\_SCREEN

Open Sat Kit

Explore cFS/SimSat Develop Apps Extend OSK

System

Start cFS Start 42 Sim Start cFS with 42 Sim

Stop cFS Stop 42 Sim

System Time(secs)

1001066

Learn OpenSatKit

Open Resource About

Learn cFS

Open Resource About

SimpleSat Ref Mission

Run Script About

Config System

Send Config Cmd Enable Telemetry

Applications

Runtime Environment	Data/File Mgmt	Autonomy
Attitude Det/Ctrl	Health Safety	Maintenance

core Flight Executive (cFE)

Event Service	Executive Service	Software Bus
Table Service	Time Service	cFE Users Guide

Flight Event Messages

Major Frame Sync too noisy (Slot 1). Disabling synchronization.



- **Classroom demonstrations & interactive exercises**
  - Target audience includes range of grade levels without software development experience
  - Software-only or COSMOS connected to a hardware platform
  - In person or students access a remote system
- **Technology development**
  - Intended for software engineers with wide range of flight or ground applications
    - Distributed systems, AI, autonomy, security, etc
- **What OSK configurations can meet these needs?**
  - Current VM distribution
  - Low-cost low-assembly hardware platform
  - Cloud-based remote access





# Tune, Verify, & Validate



CFS\_KIT PERF\_MON\_SCREEN

## Performance Monitor

**Commands**

Set Filter Mask	Set Trigger Mask
Start Data Collect	Stop Data Collect
Get File	Launch Analysis Tool

**Status**

State  Mode  Trigger Count

**Masks**

Filter

Trigger

**Log Stats**

Start  End

Count  Remaining to Write

**File Transfer**

Put File	Get File
----------	----------

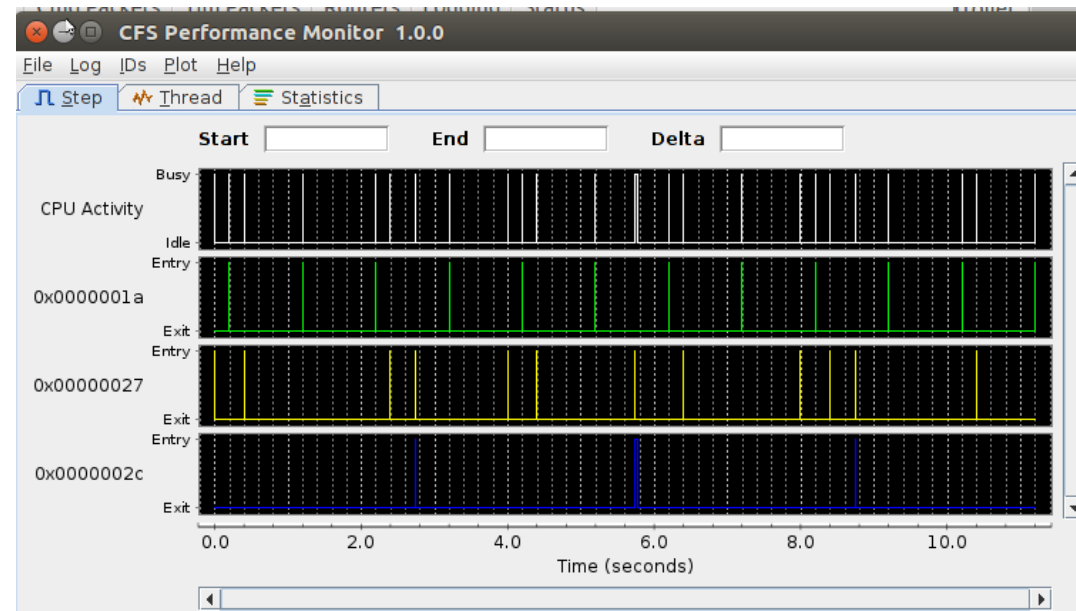
PUT\_FILE\_COUNT:  GET\_FILE\_COUNT:

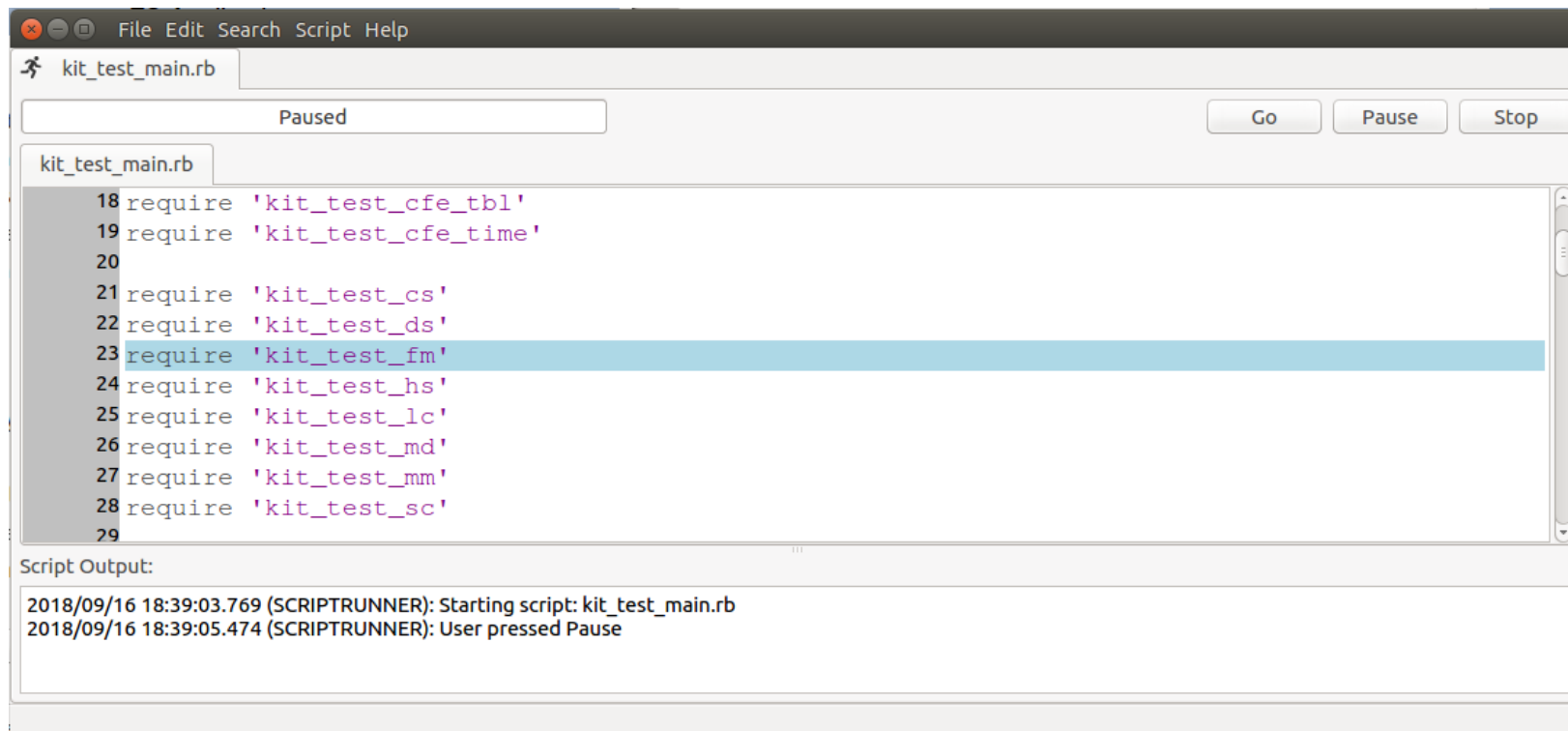
Ground Working Directory

Flight Working Directory

**Flight Event Messages**

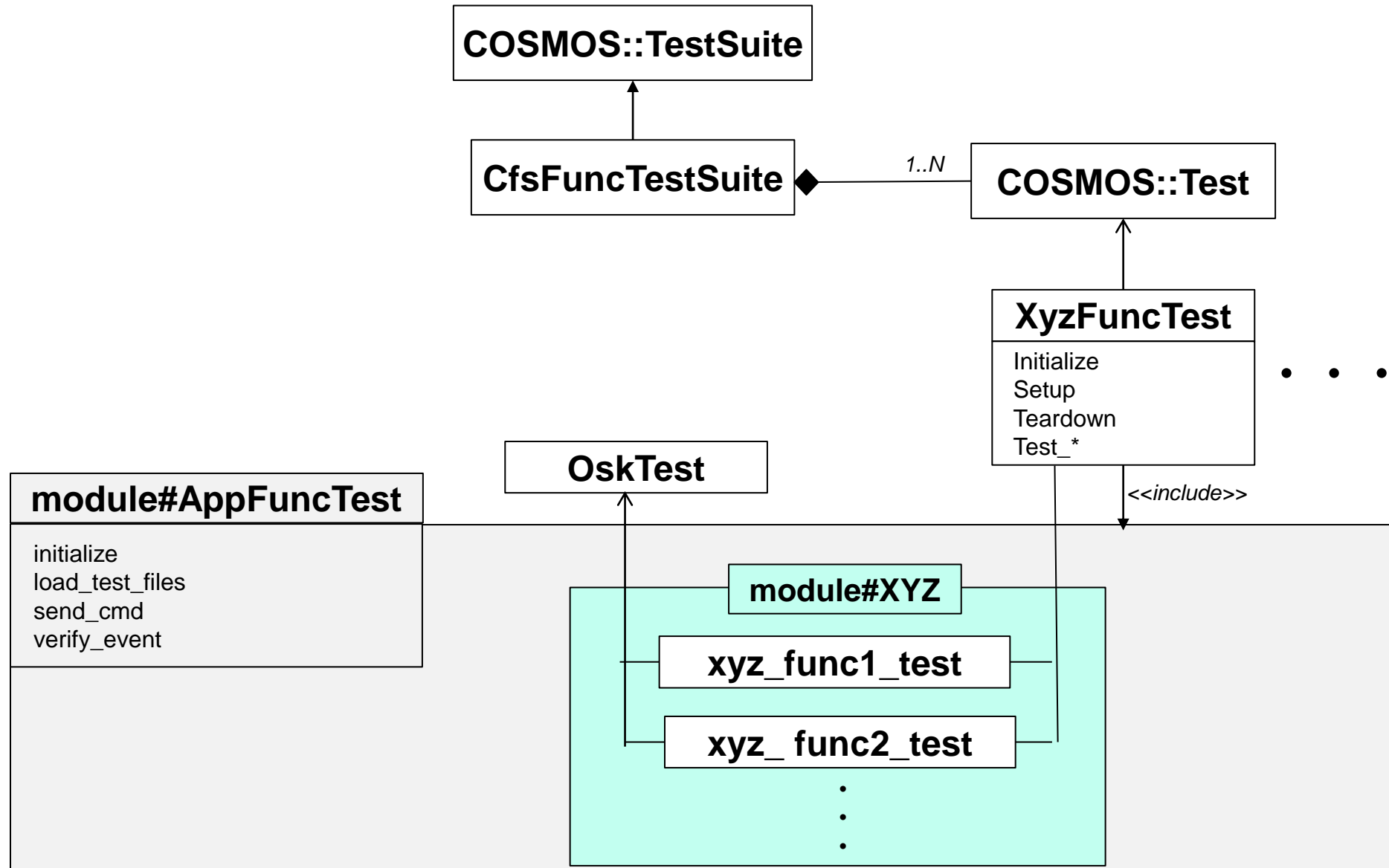
- Capture FSW performance data using screen
- Download file and <Launch Analysis Tool>





- Runs test script using Script Runner
- Issues Noop command to every application and verifies telemetry response







# SimSat Operation Script Example





# Running with 42 Simulator



**Needs updates since v2.4**  
**Merge with apply systems engineering  
processes**

CFS\_KIT SIM\_42\_SCREEN

### 42 Simulation

<b>Run 42 Sim</b>	Reconnect 42	Disconnect 42
Set Whl Tgt Mom	Manage Ctrl Tbl	Config SunValid

**I42**

Cmd Valid: 0

Cmd Error: 0

42 Connected: FALSE

42 Cycles: 0

Sensor Pkts: 0

Actuator Pkts: 0

**F42**

Cmd Valid: 0

Cmd Error: 0

Control Exec Cnt: 0

Sun Valid: 0

OVR Sun Valid: SE\_42\_SIM

#### Attitude Control

Att Err X: 0.000000	Att Err Y: 0.000000	Att Err Z: 0.000000	Plot
Wheel 1 Cmd: 0.000000	Wheel 2 Cmd: 0.000000	Wheel 3 Cmd: 0.000000	Plot

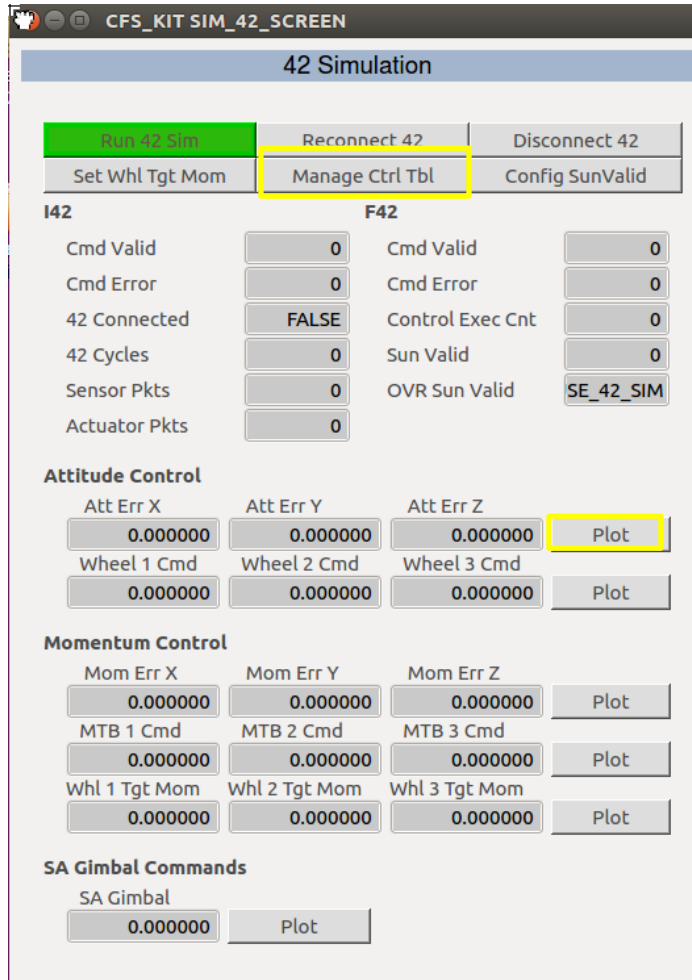
#### Momentum Control

Mom Err X: 0.000000	Mom Err Y: 0.000000	Mom Err Z: 0.000000	Plot
MTB 1 Cmd: 0.000000	MTB 2 Cmd: 0.000000	MTB 3 Cmd: 0.000000	Plot
Whl 1 Tgt Mom: 0.000000	Whl 2 Tgt Mom: 0.000000	Whl 3 Tgt Mom: 0.000000	Plot

#### SA Gimbal Commands

SA Gimbal: 0.000000	Plot
---------------------	------

- Select *<Run 42 Sim>* which will start the 42 simulator in a new terminal window.
- The 42 configuration files used in the simulation are located in directory *OpenSatKit/42/OSK*
- The simulation takes a while to initialize



- From the kit main page on the previous slide select <42 Simulator> and the screen to the left will appear.
- The 2nd row of buttons allow you to change the behavior of the control algorithms running in the FSW and are described on the next slides
- Before running the sim you will open some additional windows that will be used for your class exercise
  - Manage Control Table
  - Plot Attitude Errors



F42 TBL\_SCR

F42 Control Table

Get Current Values Load Screen Values Restore Defaults

PD Gain Parameter

W Z

Wheel Target Momentum Limits

Lower Upper

Moment of Inertia

X Y Z

Control Gains

	X-Axis	Y-Axis	Z-Axis
Kr	0.006287	0.087920	0.098000
Kp	0.000449	0.006280	0.007000

- Selecting *<Manage Control Table>* on the 42 Sim screen produces the screen to the left.
- Select *<Get Current Values>* and it will populate the screen with the current control table values. This takes a little time because it is transferring a file from flight to ground
- Edit the screen as desired and click *<Load Screen Values>* to replace the current control table values
- The defaults can be restored by clicking *<Restore Defaults>*

F42 TBL\_SCR

F42 Control Table

Get Current Values Load Screen Values Restore Defaults

PD Gain Parameter

W Z

0.628 0.7

Wheel Target Momentum Limits

Lower Upper

-0.9 0.9

Moment of Inertia

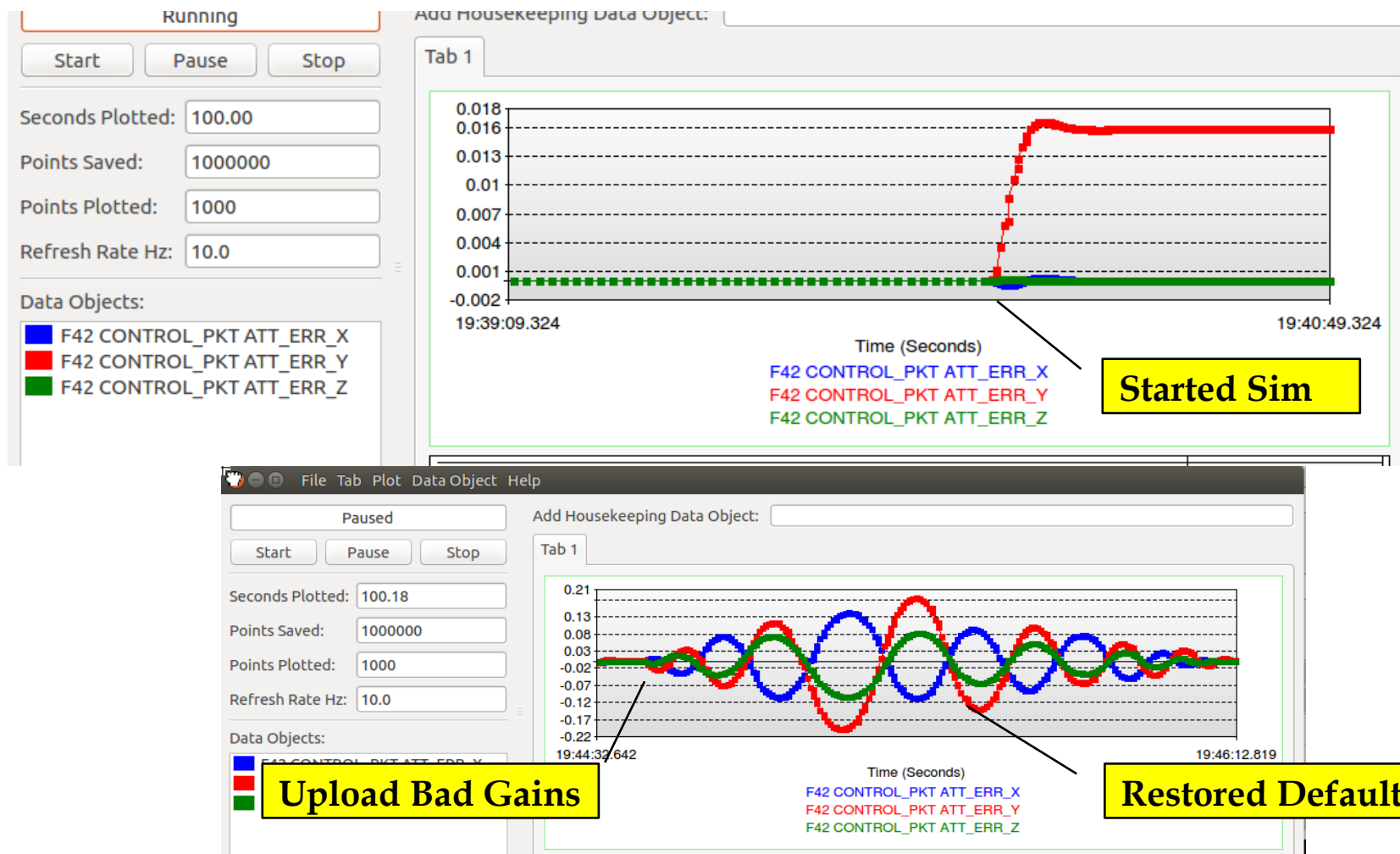
X Y Z

0.119835 0.14778 0.044908

Control Gains

	X-Axis	Y-Axis	Z-Axis
Kr	0.105359	0.129928	0.039483
Kp	0.047261	0.058282	0.017711

- Selecting <Plot> button next to the attitude errors produces the screen below



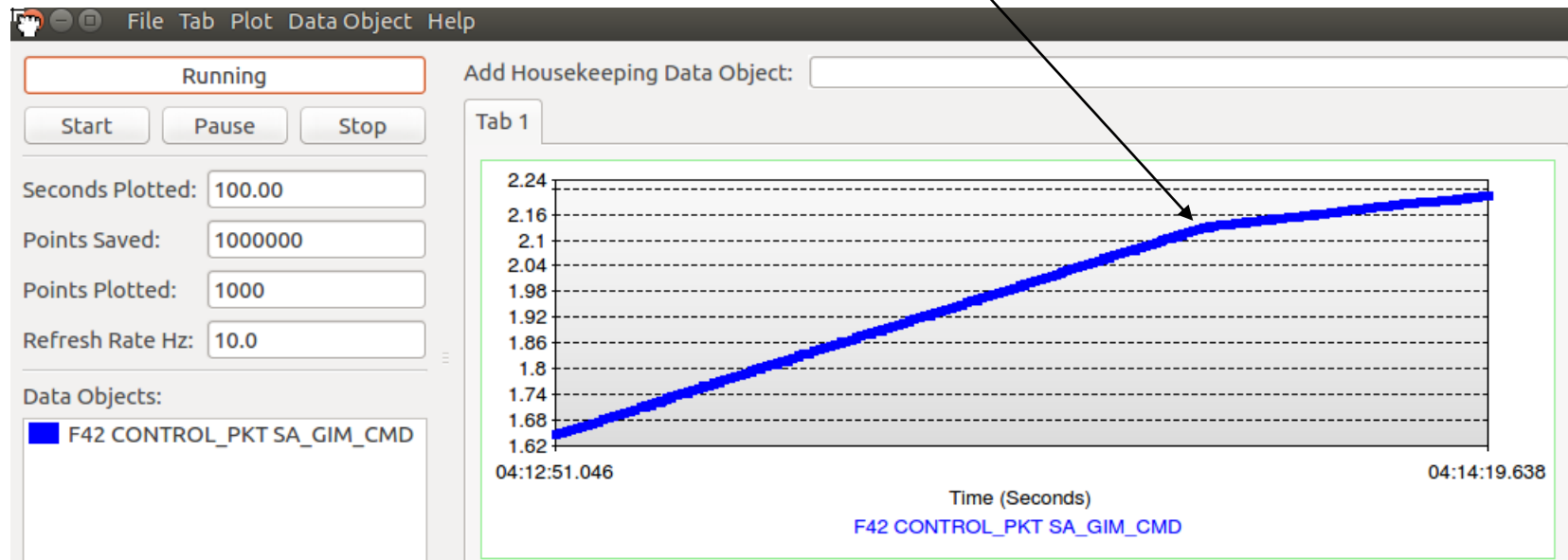


# Additional Configuration Options



- **The kit includes two additional configuration options that can be manipulated**
  1. Wheel target Momentum
  2. Sun Valid Configuration

- Selecting <Config SunValid> to override the current sun valid flag
- The plot below shows gimbal command
  - The linear portion had a valid sun and the bend occurred when the SunValid was overridden to false.





# Sim Termination



1. Click <*Disconnect 42*> to end a 42 simulation that is running with the FSW
2. To terminate the flight software click on the terminal window with the FSW messages and then enter ctrl-c
3. Each of the cosmos windows will need to be closed individually. If you close the COSMOS TlmViewer window first it prompt you to close all of the telemetry screens at once.



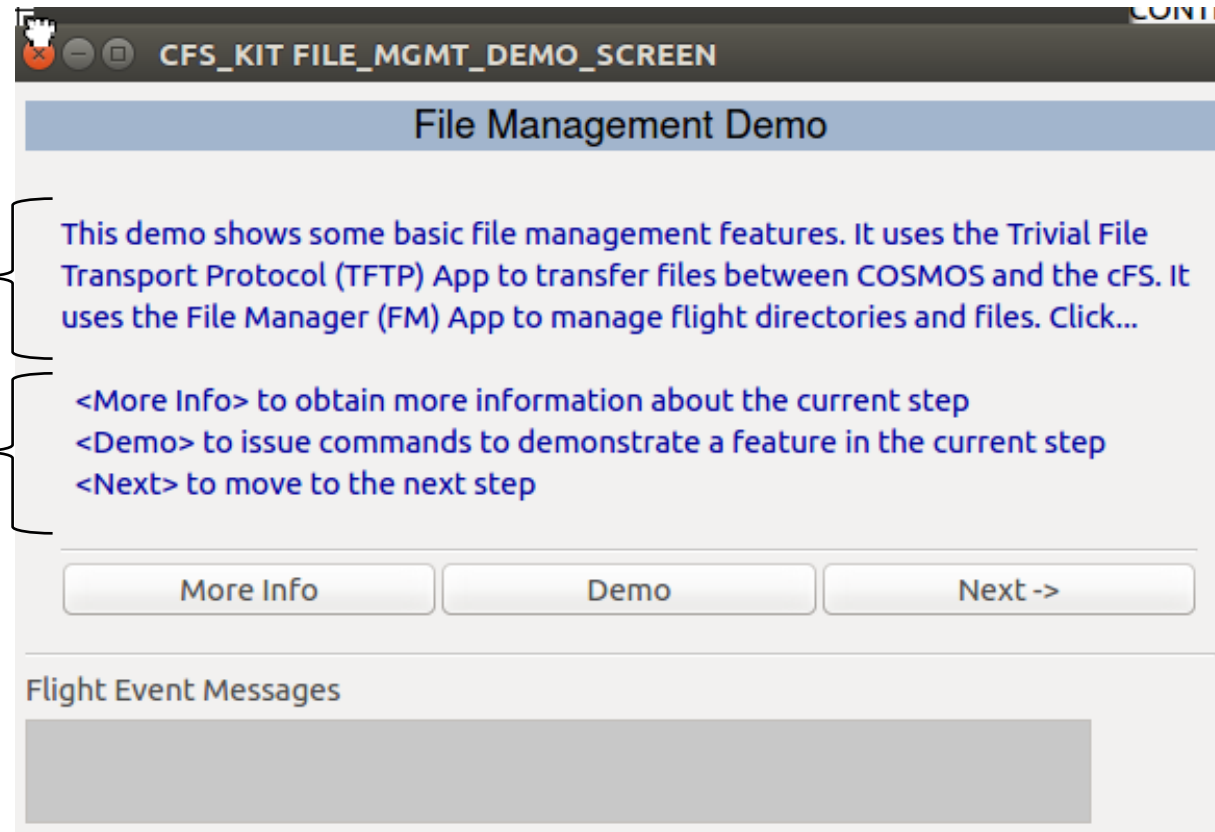


# Demos

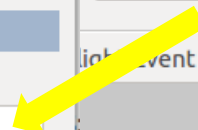
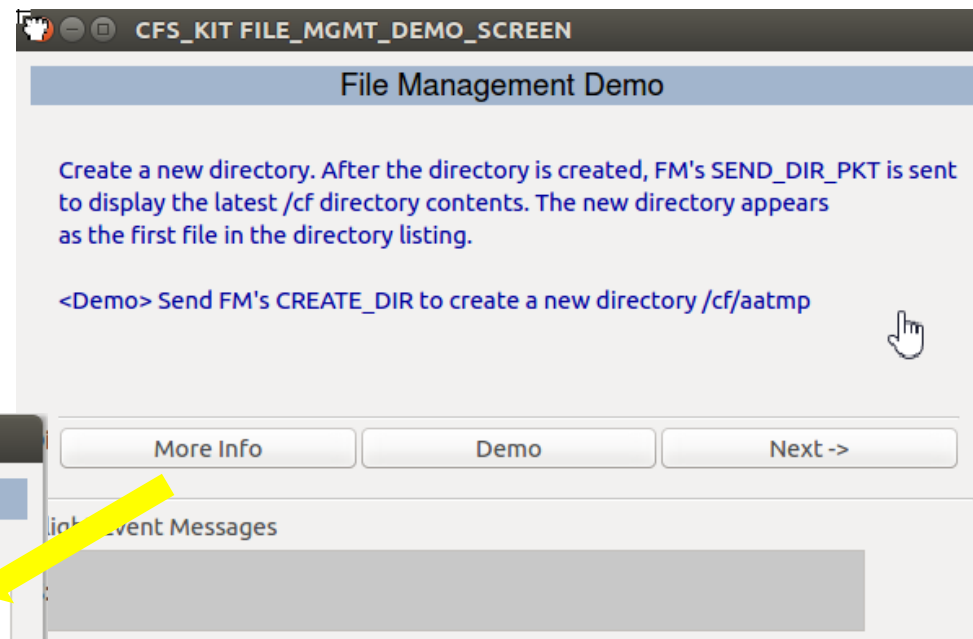
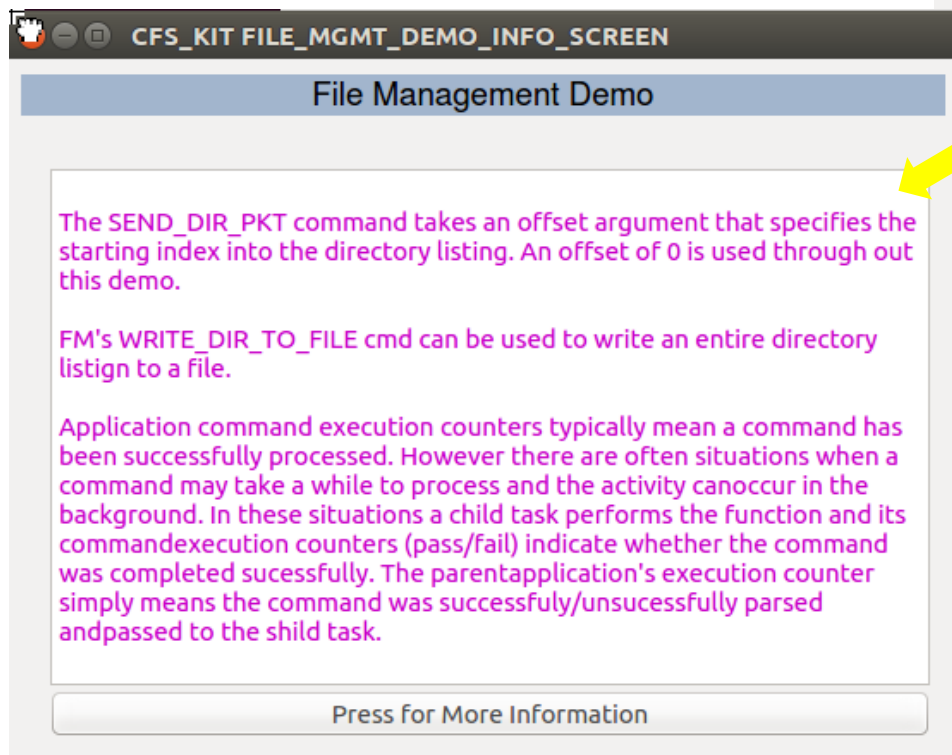
Each demo follows a common user screen configuration

Description of  
current step

Button usage  
description



**<More Info> provides detailed context-specific information**







# Application Functional Screens

CFS\_KIT FILE\_MGMT\_SCREEN

## File Management

### Directory Management

Create	Delete
List to Packet	Write to File

### File Manager Directory Listing

DIRNAME:

TOTALFILES:

PACKETFILES:

FIRSTFILE:

FILE01\_NAME:

FILE02\_NAME:

FILE03\_NAME:

FILE04\_NAME:

FILE05\_NAME:

FILE06\_NAME:

FILE07\_NAME:

FILE08\_NAME:

FILE10\_NAME:

FILE11\_NAME:

FILE12\_NAME:

### File Management

Copy	Move
Rename	Decompress
Delete	Delete All
Concat	Get Info
List Open	

### File Manager Housekeeping

Cmd Valid Cnt

Cmd Error Cnt

Child Cmd Valid Cnt

Child Cmd Error Cnt

### File Transfer

Put File	Get File
----------	----------

PUT\_FILE\_COUNT:  GET\_FILE\_COUNT:

Ground Working Directory

Flight Working Directory

### Event Messages

- <List to Packet> commands File Manage (FM)
  - To send a directory listing
  - The command uses a directory listing alphabetical "offset" to determine which file to start with in the listing
- OSK uses the verbs *list* and *send* to indicate information is sent in a telemetry packet.
- *Write* is used when information is written to a file
- <List to Packet> commands File Manage (FM)
  - To send a directory listing
  - The command uses a directory listing alphabetical "offset" to determine which file to start with in the listing



CFS\_KIT TABLE\_MGMT\_SCREEN

## Table Management

**Table Management**

Load Table	Validate	Activate
Abort Load	Dump Table	Display Table

**Table Registry**

Display Registry Write Registry to File

**Table Manager Housekeeping**

Cmd Valid Cnt	0
Cmd Error Cnt	0
Last Updated Table	
Last File Loaded	
Last File Dumped	
Last Table Loaded	

**Table Registry Listing**

NAME:

SIZE:  0

CRITICAL:  0

TABLE\_LOADED\_ONCE:  0

LOAD\_PENDING:  0

DUMP\_ONLY:  0

DBL\_BUFFERED:  0

LAST\_UPD\_TIME\_SECONDS:  0

FILE\_CREATE\_TIME\_SECS:  0

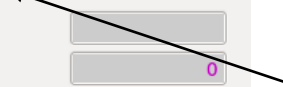
LAST\_FILE\_LOADED:

OWNER\_APP\_NAME:

**File Transfer**

Put File	Get File
PUT_FILE_COUNT: <input type="text"/> 0	GET_FILE_COUNT: <input type="text"/> 0
Ground Working Directory	
<input type="text"/>	
Flight Working Directory	
<input type="text"/>	

**Flight Event Messages**

- Load a new FSW table  
 <Put File> transfers file from ground to flight  
 <Load Table> into table buffer  
 <Validate> table via app validation function  
 <Activate> new table
- <Display Registry> sends a table's registry information in a telemetry packet
 
- Dump and display FSW table  
 <Dump Table> to onboard file  
 <Get File> transfers file from flight to ground  
 <Display Table> launches COSMOS Table Manager to view file. Requires binary file definition.

CFS\_KIT MEMORY\_MGMT\_SCREEN

## Memory Management

### Memory Manager

Lookup Symbol	Peek Address	Poke Address	Dump to Event
Fill Block	Load from File	Dump to File	Manage Checksums

### Memory Dwell

Start Dwell	Stop Dwell
Jam Dwell Tbl	Dwell Tbl 1 Pkt

### Memory Dwell Housekeeping

Cmd Valid Cnt	0
Cmd Error Cnt	0
Enable Mask	0000

### Memory Manager Status

Cmd Valid Cnt	0
Cmd Error Cnt	0
Last Action	NONE
Mem Type	0
Address	00000000
Fill Pattern	00000000
Bytes Processed	0

Last Memory Manager File

File Transfer

Put File	Get File
PUT_FILE_COUNT: 0	GET_FILE_COUNT: 0
Ground Working Directory	
Flight Working Directory	

Flight Event Messages

- Memory Manager (MM) and Memory Dwell (MD) apps are typically used for inflight maintenance.
- MM commands allow direct access to any memory location
- MD generates telemetry packets that contain the contents of table-specified memory locations
  - Only 1 dwell table telemetry packet is defined
  - *<Jam Dwell Table>* allows the dwell table to be loaded without using the table load service
- The FSW can easily be corrupted using memory manager
- The memory management demo is a good place to start since it demonstrates MM and MD using safe memory locations



# Recorder Management



CFS\_KIT RECORDER\_MGMT\_SCREEN

Recorder Management

Data Storage App Status

Enable/Disable

Dest File 1..4 Info

Dest File 5..8 Info

Cmd Valid Cnt

0

Cmd Error Cnt

0

State

0

Set Destination File Configuration

Enable/Disable

Sequence Count

Filename Type

File Path Name

File Base Name

File Extension

Max File Size

Max File Age

Close 1/All Files

Tbl Load Count

0

Tbl Access Err Cnt

0

File Write Valid Cnt

0

File Write Invalid Cnt

0

Hdr Update Valid Cnt

0

Hdr Update Invalid Cnt

0

Set Packet Filter Configuration

Dest File

Add Message

Algorithm

Filter Type

Tbl Load Cnt

0

Tbl Access Err Cnt

0

Pkt Discard Cnt

0

Pkt Ignored Cnt

0

Pkt Filtered Cnt

0

Pkt Stored Cnt

0

Packet Filter File

File Transfer

Put File

Get File

PUT\_FILE\_COUNT:

0

GET\_FILE\_COUNT:

0

Ground Working Directory

Flight Working Directory

Flight Event Messages

Mission FSW Quick Start Guide

Page 62



# Autonomy Management



CFS\_KIT AUTONOMY\_MGMT\_SCREEN

Autonomy Management

Stored Command(SC) App - Relative Time Sequences(RTS)

Start RTS	Stop RTS	Enable RTS	Disable RTS
Start Group	Stop Group	Enable Group	Disable Group
Cmd Valid Cnt	0	Cmd Error Cnt	0

RTS Status

RTS	64 .. 49	48 .. 33	32 .. 17	16 .. 1
EXECUTING	0000	0000	0000	0000
DISABLED	0000	0000	0000	0000

Start Cnt

0000

Start Err Cnt

0000

Next Time

0000000

Active Cnt

0000

Next RTS Num

0000

RTS CMD Cnt

000000

CMD Err Cnt

0000

Err RTS#

0000

Err RTS Offset

0000

Limit Checker(LC) App

Reset WP Stats	Reset AP Stats	Set AP State	Set AP Prem Off
Set App State	App State	0	
Cmd Valid Cnt	0	Cmd Error Cnt	0

Watch Points(WP) Action Points(AP) Status

Watch Points (2-bits per WP)

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

Action Point (4-bits per AP)

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

PASS RTS EXE Cnt

0

RTS EXE Cnt

0

WPs in Use

0

WP MSG Mon Cnt

0

Active APs

0

AP Sample Cnt

0

Flight Event Messages

