OpenSatKit Quick Start

v2.6

December 2020

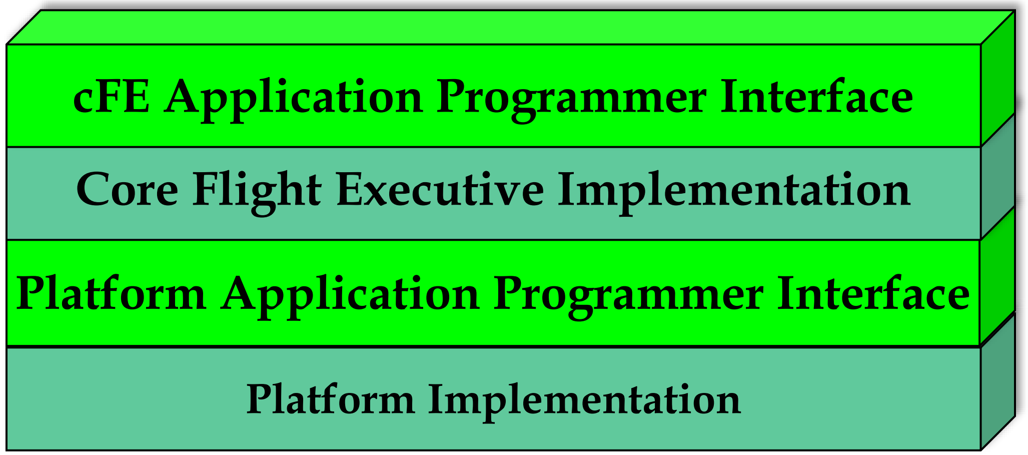
Introduction

- **The primary objectives of OpenSatKit (OSK) are to**
  - Provide a core Flight System (cFS) training environment
  - Provide a cFS application development environment
  - Serve as a starting point for a new cFS-based project
- **The cFS is an open architecture that is designed to be ported and extended**
  - These attributes add end-user deployment/configuration complexity
  - OSK provides fully functional cFS system deployed on Linux, however…
- **OSK introduces additional complexity because it integrates two additional powerful software packages, COSMOS and the 42 Simulator, that have their own learning curve.**

The cFS provides high quality flight heritage software that implements a significant amount of mission functionality so the rewards are high if you can persist through the learning curve!

cFS Introduction

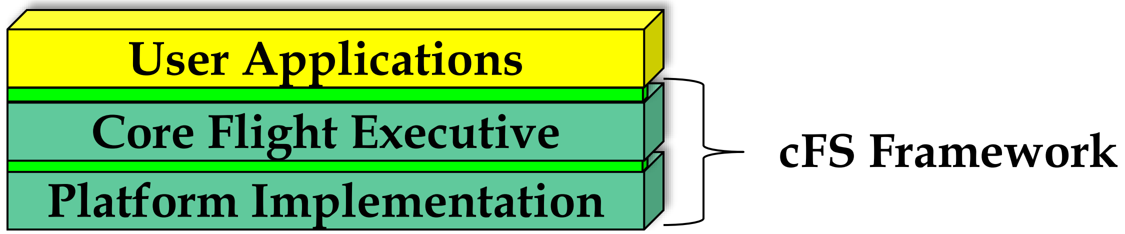
- A NASA multi-center configuration controlled open source flight software framework



cFE Application Programmer Interface
Core Flight Executive Implementation
Platform Application Programmer Interface
Platform Implementation

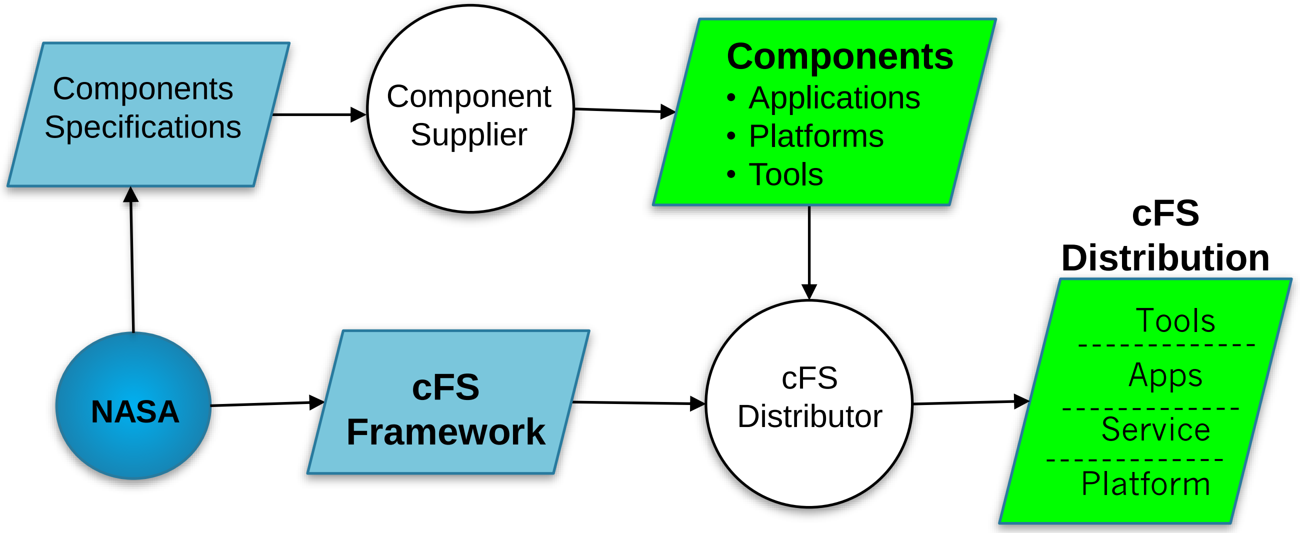
- Layered architecture with international standards-based interfaces
- Provides development tools and runtime environment for user applications
- Reusable NASA Class A/B lifecycle artifacts: requirements, design, code, tests, and documents

- The framework is ported to a platform and augmented with applications to create Core Flight System (cFS) distributions



- A worldwide community from government, industry, and academia

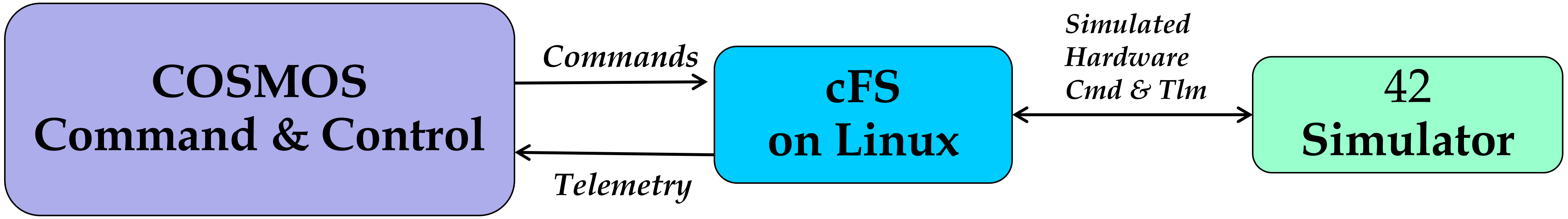
cFS Product Model



- **A NASA multi-center configuration control board (CCB) manages releases of the open source cFS Framework and component specifications**

- **Community members (regardless of affiliation)**
  - Supply applications, platforms, and tools
  - Create cFS distributions – OSK is a distribution

OSK Architecture



- **In addition to the cFS itself, OSK uses two additional open source applications**
  - Ball Aerospace's COSMOS command and control platform for embedded systems
  - NASA Goddard's 42 dynamic simulator
- **Each open source package is contained in its own OpenSatKit subdirectory**

OSK User Objectives

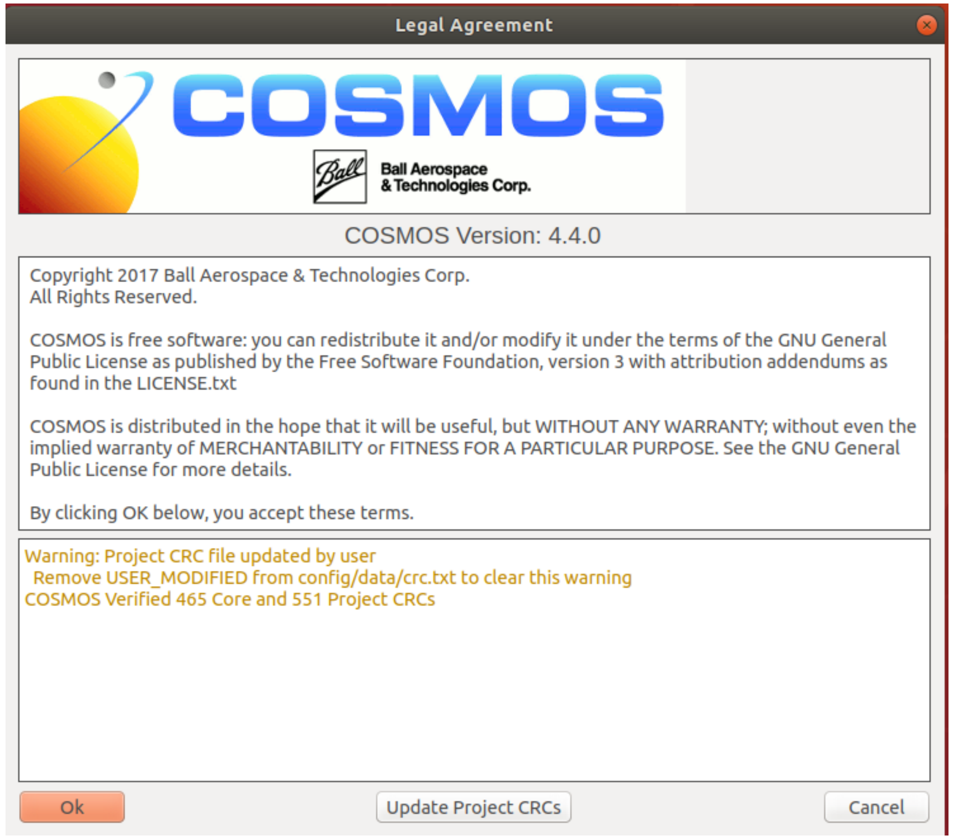
1. **Learn the cFS**
   - cFE services
   - cFS apps
   - cFS system
2. **Manage and develop applications within the Linux desktop environment**
   - Create new apps
   - Import community apps
3. **Extend OSK**
   - **Evolve the default system to a user system**
   - **Deploy the cFS to a target system**
     - Use OSK as a ground system for a remote system
     - Run benchmarks
   - **Develop advanced applications**
     - E.g. External Code Interface (ECI)
   - **Create bridges to other systems**
     - E.g. Robot Operating System (ROS), openMCT

OSK Strategy

- **OSK implements extensive COSMOS configurations and customizations so OSK's screens can serve as the primary user interface for the goals listed below**
  - Doesn't preclude direct use of COSMOS tools
- **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
- **Organize screens and content to align with OSK user objectives**
- **Learning resources use a combination of documents, screen and script based demos, and links to videos**

Running OSK (1 of 2)

- Open a terminal window (Ctrl-Alt-t)
- 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-master/cfs" if OpenSatKit was installed in the home directory for an account named "user"
- Change directory to cosmos
  - cd ~/cosmos
- Start COSMOS
  - ~/cosmos$ ruby Launcher
  - You'll see a screen similar to the right.
    - Select <OK>
    - This creates the "Launcher" screen shown on the next slide



COSMOS Version: 4.4.0

Copyright 2017 Ball Aerospace & Technologies Corp.
All Rights Reserved.

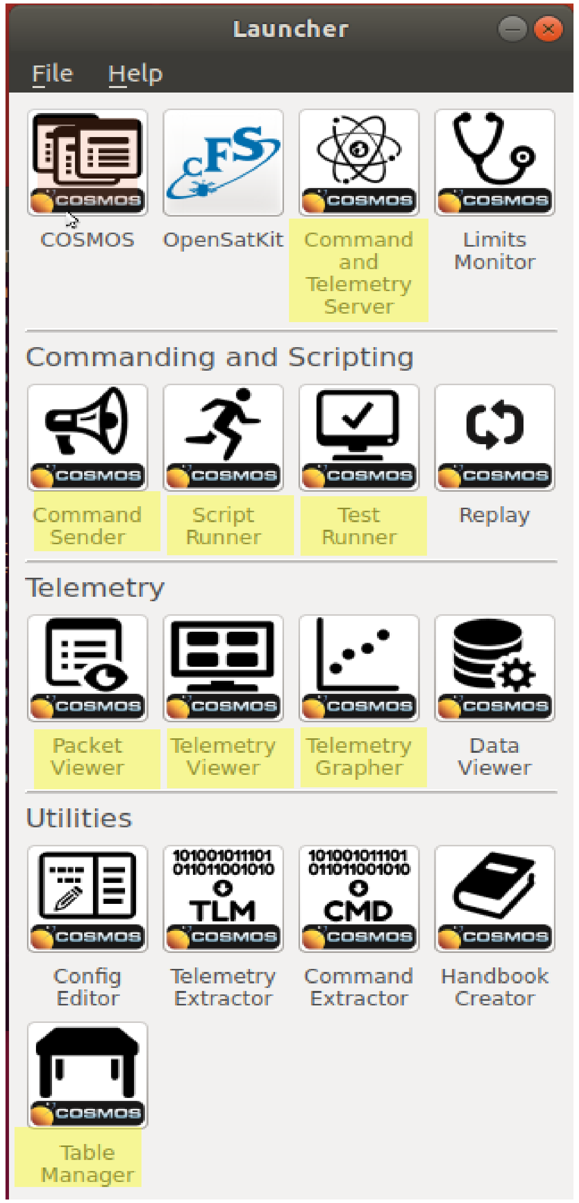
COSMOS is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 with attribution addendums as found in the LICENSE.txt

COSMOS is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

By clicking OK below, you accept these terms.

```
Warning: Project CRC file updated by user
 Remove USER_MODIFIED from config/data/crc.txt to clear this warning
COSMOS Verified 465 Core and 551 Project CRCs
```

Running OSK (2 of 2)

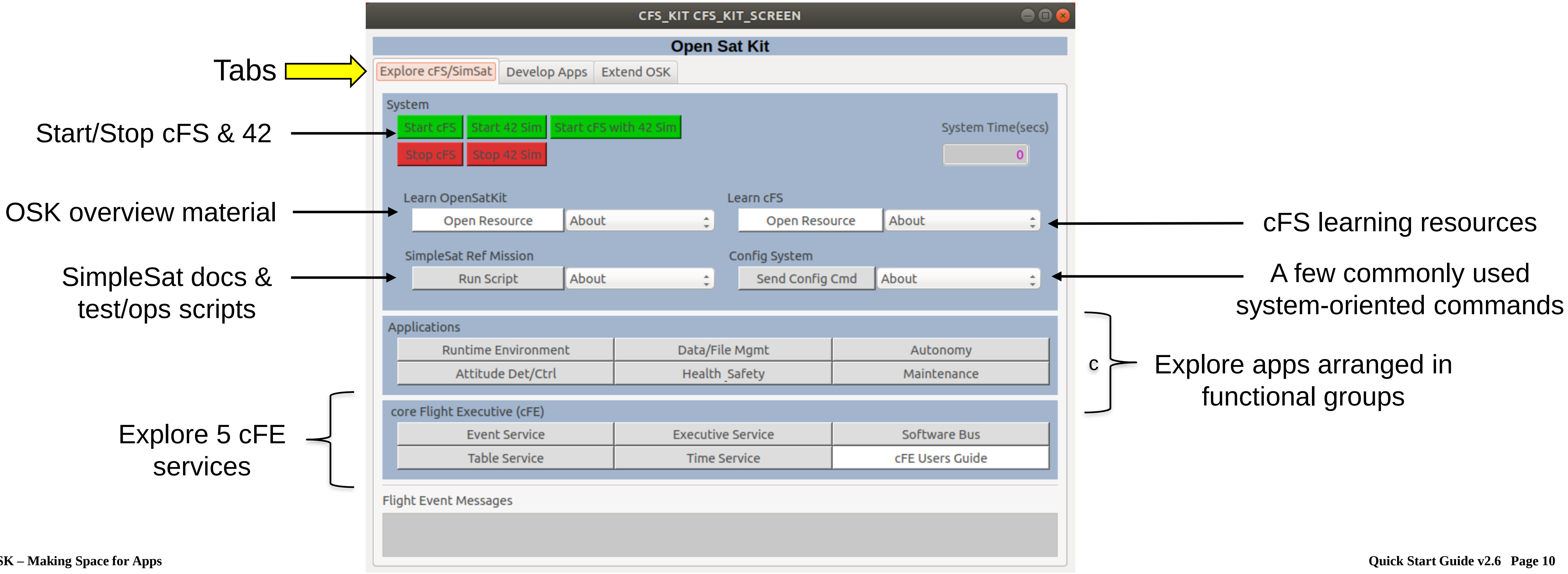


- **Each tools on the COSMOS "Launcher" runs as a separate Linux process with a Graphical User Interface (GUI)**
- **Shaded tool titles indicate the COSMOS tools** used by OSK**
  - You do not have to invoke these tools directly
  - OSK screens launch COSMOS tools as they are needed to perform a task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

** See COSMOS Appendix for a brief description of each tool

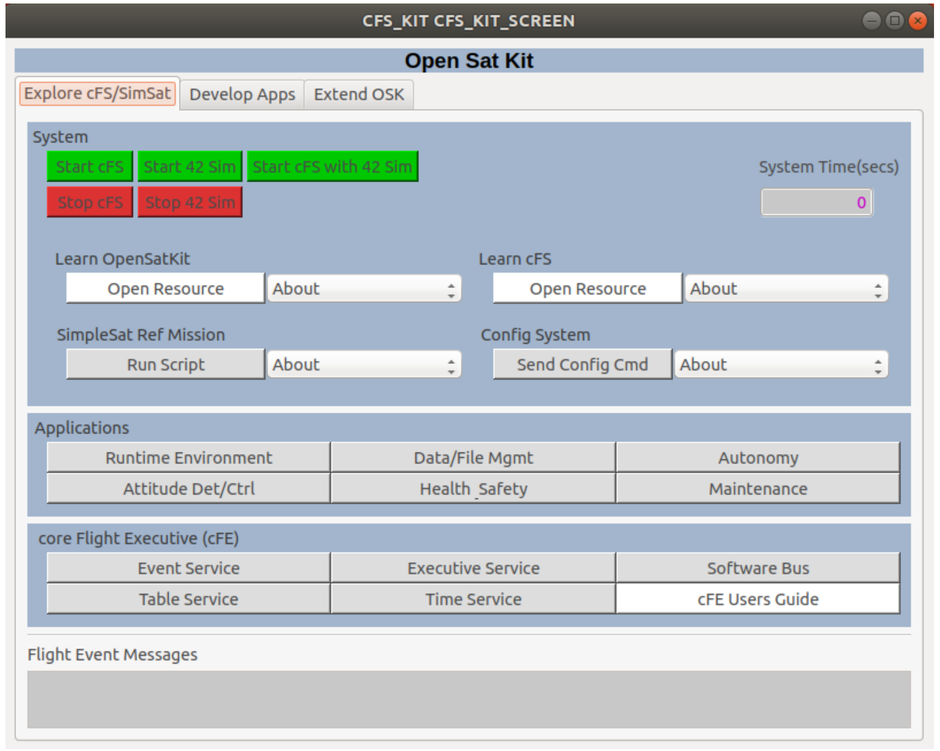
OSK Main Screen

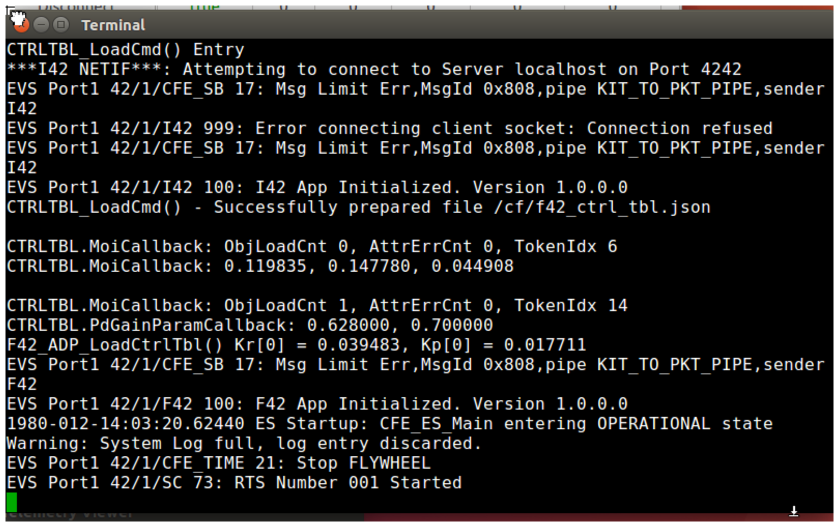
- Three tabs ***Explore cFS/SimSat***, ***Manage Apps***, and ***Extend OSK*** provide the top-level organization
- ***Explore cFS/SimSat*** allows the user to learn the cFS using SimSat
- ***Manage Apps*** provides tools for adding, removing, and creating apps
- ***Extend OSK*** is in its infancy, but it's goal is to allow the user to bridge the cFS to other systems and control remote devices



Start the Flight Software (FSW)

- **Click <Start cFS> to run the FSW. <Start cFS/42> is used later.**
  - A new terminal window is created for the Linux process running the cFS
  - Enter your user account password when prompted for a password.
- **In a few seconds the System Time box should turn white time with advancing**
  - If time doesn't advance select <Enable Tlm> under "Config System"





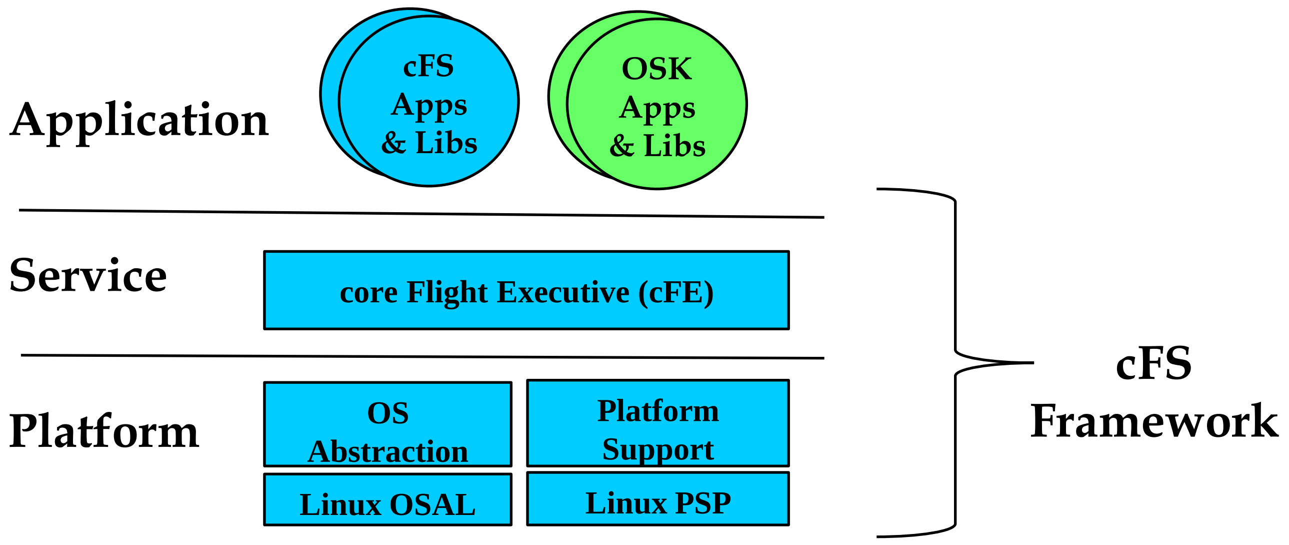
```
CTRLTBL_LoadCmd() Entry
***I42 NETIF***: Attempting to connect to Server localhost on Port 4242
EVS Port1 42/1/CFE_SB 17: Msg Limit Err,MsgId 0x808,pipe KIT_TO_PKT_PIPE,sender
I42
EVS Port1 42/1/I42 999: Error connecting client socket: Connection refused
EVS Port1 42/1/CFE_SB 17: Msg Limit Err,MsgId 0x808,pipe KIT_TO_PKT_PIPE,sender
I42
EVS Port1 42/1/I42 100: I42 App Initialized. Version 1.0.0.0
CTRLTBL_LoadCmd() - Successfully prepared file /cf/f42_ctrl_tbl.json

CTRLTBL.MoiCallback: ObjLoadCnt 0, AttrErrCnt 0, TokenIdx 6
CTRLTBL.MoiCallback: 0.119835, 0.147780, 0.044908

CTRLTBL.MoiCallback: ObjLoadCnt 1, AttrErrCnt 0, TokenIdx 14
CTRLTBL.PdGainParamCallback: 0.628000, 0.700000
F42_ADP_LoadCtrlTbl() Kr[0] = 0.039483, Kp[0] = 0.017711
EVS Port1 42/1/CFE_SB 17: Msg Limit Err,MsgId 0x808,pipe KIT_TO_PKT_PIPE,sender
F42
EVS Port1 42/1/F42 100: F42 App Initialized. Version 1.0.0.0
1980-012-14:03:20.62440 ES Startup: CFE_ES_Main entering OPERATIONAL state
Warning: System Log full, log entry discarded.
EVS Port1 42/1/CFE_TIME 21: Stop FLYWHEEL
EVS Port1 42/1/SC 73: RTS Number 001 Started
```

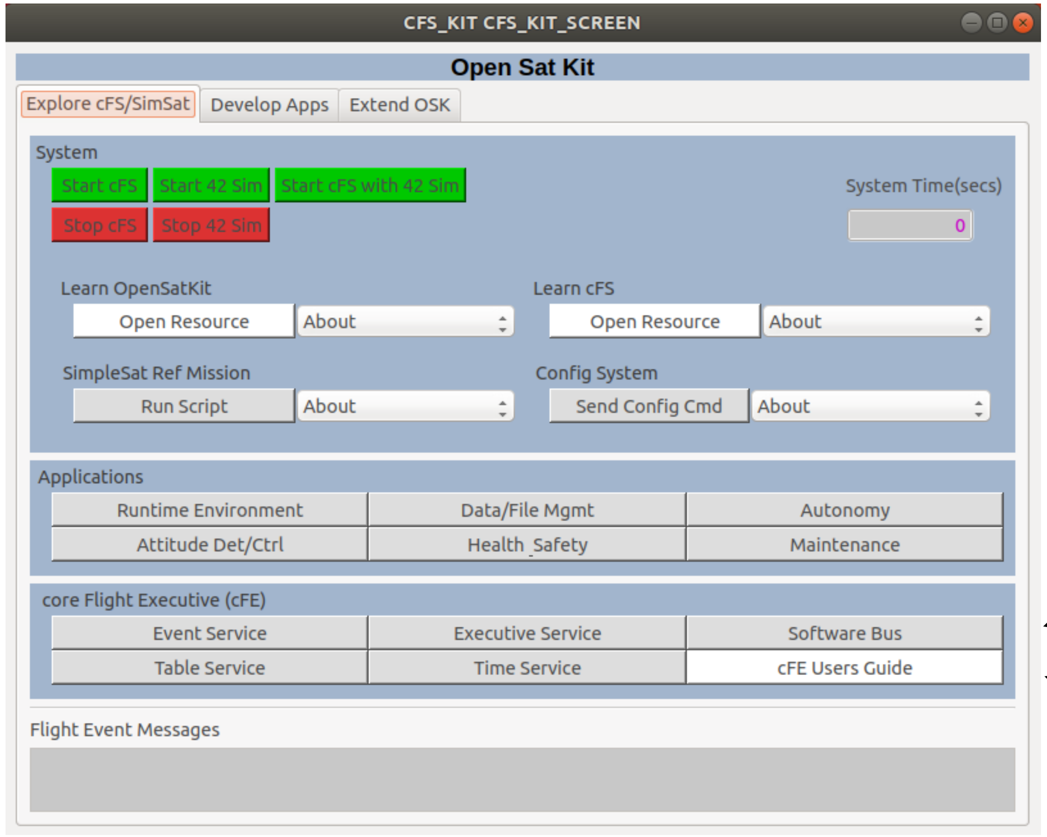
What Just Happened?

- The <Start cFS> button invoked a ruby script that created a new terminal window executing the "cFS Framework"
- The cFS Framework is the bottom two layers of the 3-tiered cFS architecture. It is a portable application runtime environment that uses a startup script (cfe_es_startup.scr) to determine which apps to load during initialization. OSK's startup script is configured for SimSat.



Core Flight Executive (cFE)

- The cFE has 5 services
  - **Executive Services (ES):** Manage the embedded software system and create an application runtime environment
  - **Time Services (TIME):** Manage spacecraft time
  - **Event Services (EVS):** Provide a service for sending, filtering, and logging event messages (time stamped text messages).
  - **Software Bus (SB) Services:** Provide an application publish/subscribe messaging service
  - **Table Services (TBL):** Manage application binary file table images

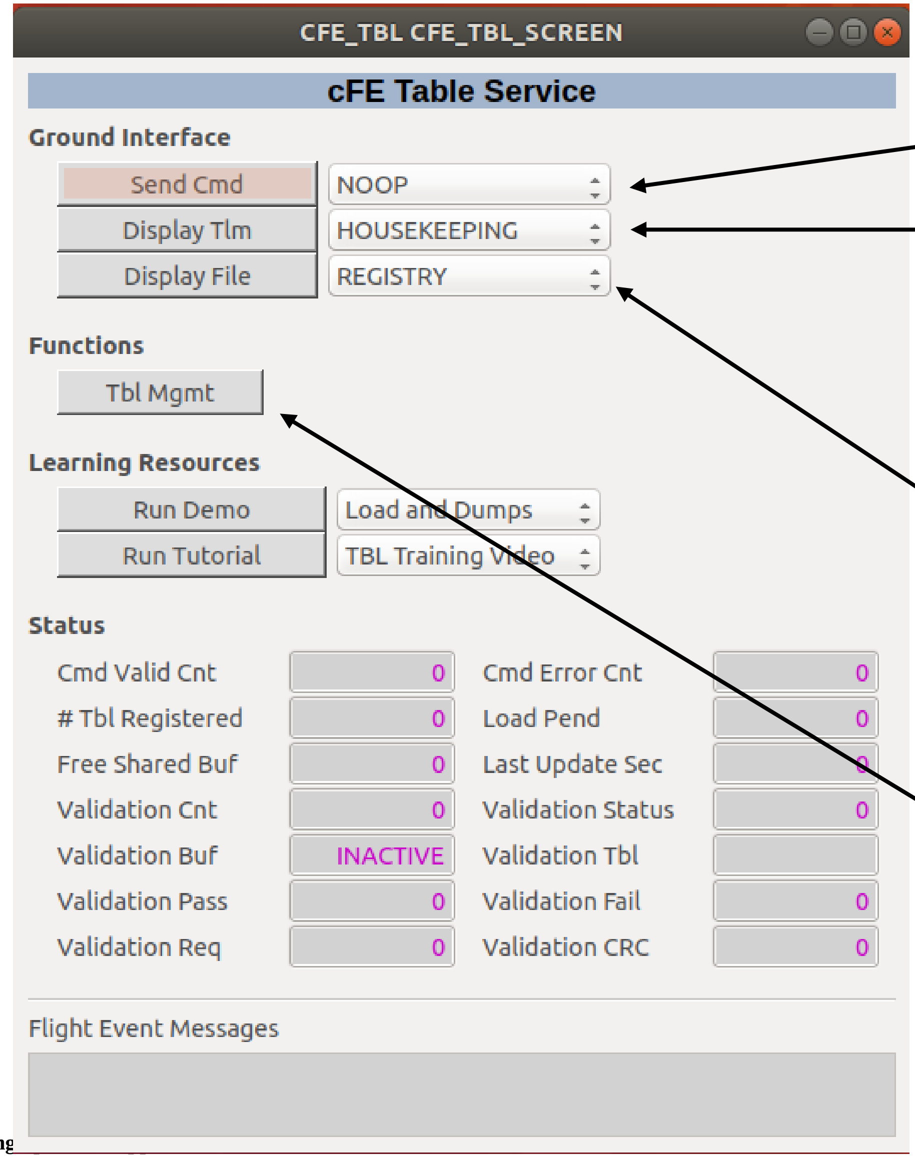


One button/screen for each service

cFE HTML User'sGuide

cFE Service Screen (1 of 2)

Table Service screen shown. All cFE screens have the same layout but may not have every component/button


CFE_TBL CFE_TBL_SCREEN

cFE Table Service

Ground Interface

Send Cmd	NOOP
Display Tlm	HOUSEKEEPING
Display File	REGISTRY

Functions

Tbl Mgmt

Learning Resources

Run Demo	Load and Dumps
Run Tutorial	TBL Training Video

Status

Cmd Valid Cnt	0	Cmd Error Cnt	0
# Tbl Registered	0	Load Pend	0
Free Shared Buf	0	Last Update Sec	0
Validation Cnt	0	Validation Status	0
Validation Buf	INACTIVE	Validation Tbl	
Validation Pass	0	Validation Fail	0
Validation Req	0	Validation CRC	0

Flight Event Messages

Select and send commands

Display a telemetry packet using COSMOS's Packet Viewer.

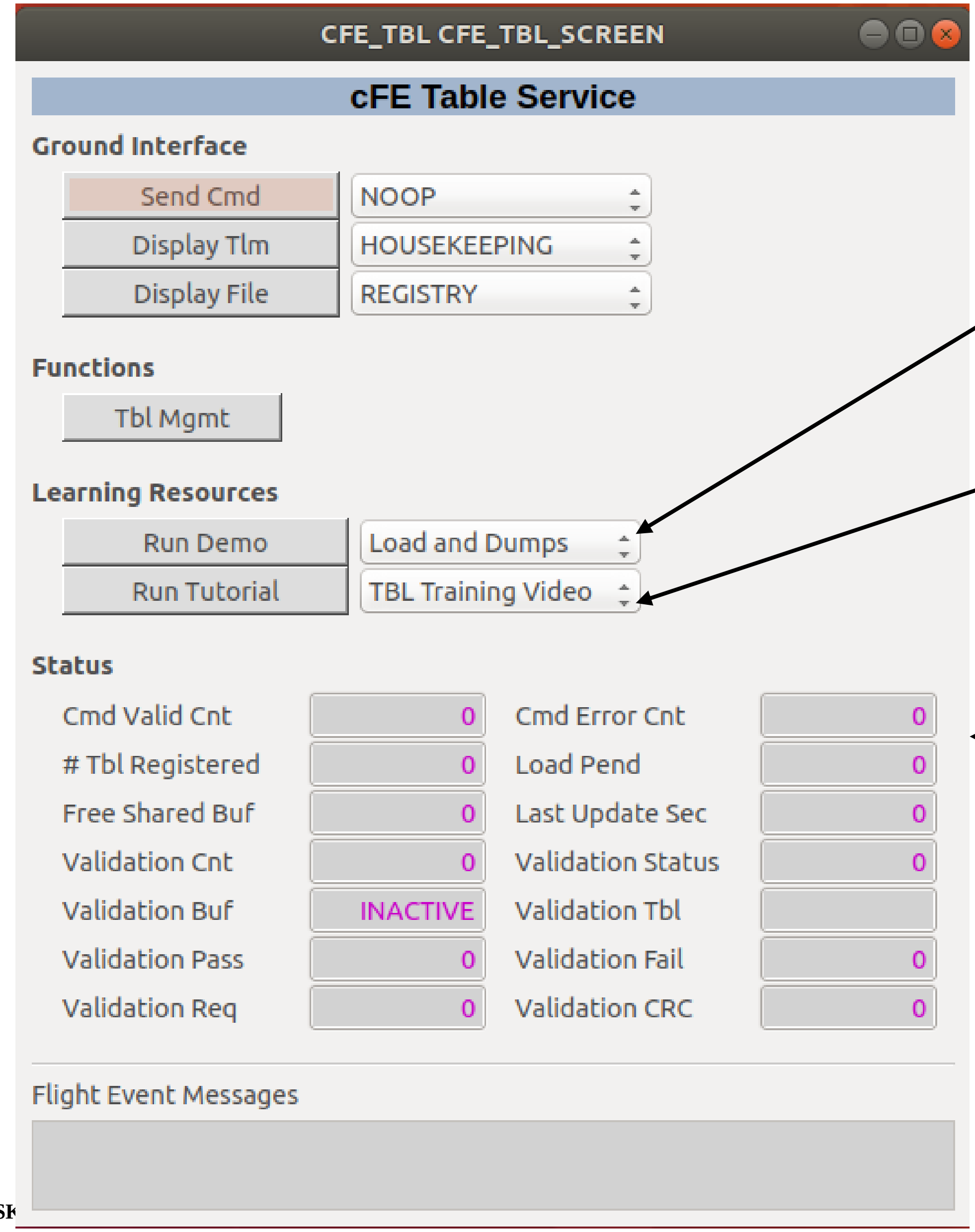
- Telemetry packets can be generated in response to a command
- E.g. Telemeter the registration information for a single table

Display a binary file using COSMOS's Table Manager

- Binary files can be generated in response to a command.
- E.g. Dump the entire table registry to a file

Display a screen that simplifies user interaction with a service

cFE Service Screen (2 of 2)



cFE Table Service

Ground Interface

Send Cmd	NOOP
Display Tlm	HOUSEKEEPING
Display File	REGISTRY

Functions

Learning Resources

Run Demo	Load and Dumps
Run Tutorial	TBL Training Video

Status

Cmd Valid Cnt	0	Cmd Error Cnt	0
# Tbl Registered	0	Load Pend	0
Free Shared Buf	0	Last Update Sec	0
Validation Cnt	0	Validation Status	0
Validation Buf	INACTIVE	Validation Tbl	
Validation Pass	0	Validation Fail	0
Validation Req	0	Validation CRC	0

Flight Event Messages

Select and run a demo

- Demos are a sequence of interactive screens that step the user through a task

Select and run a tutorial

- Tutorial are typically, but not limited to a set of slides coupled with a ruby script for exercises

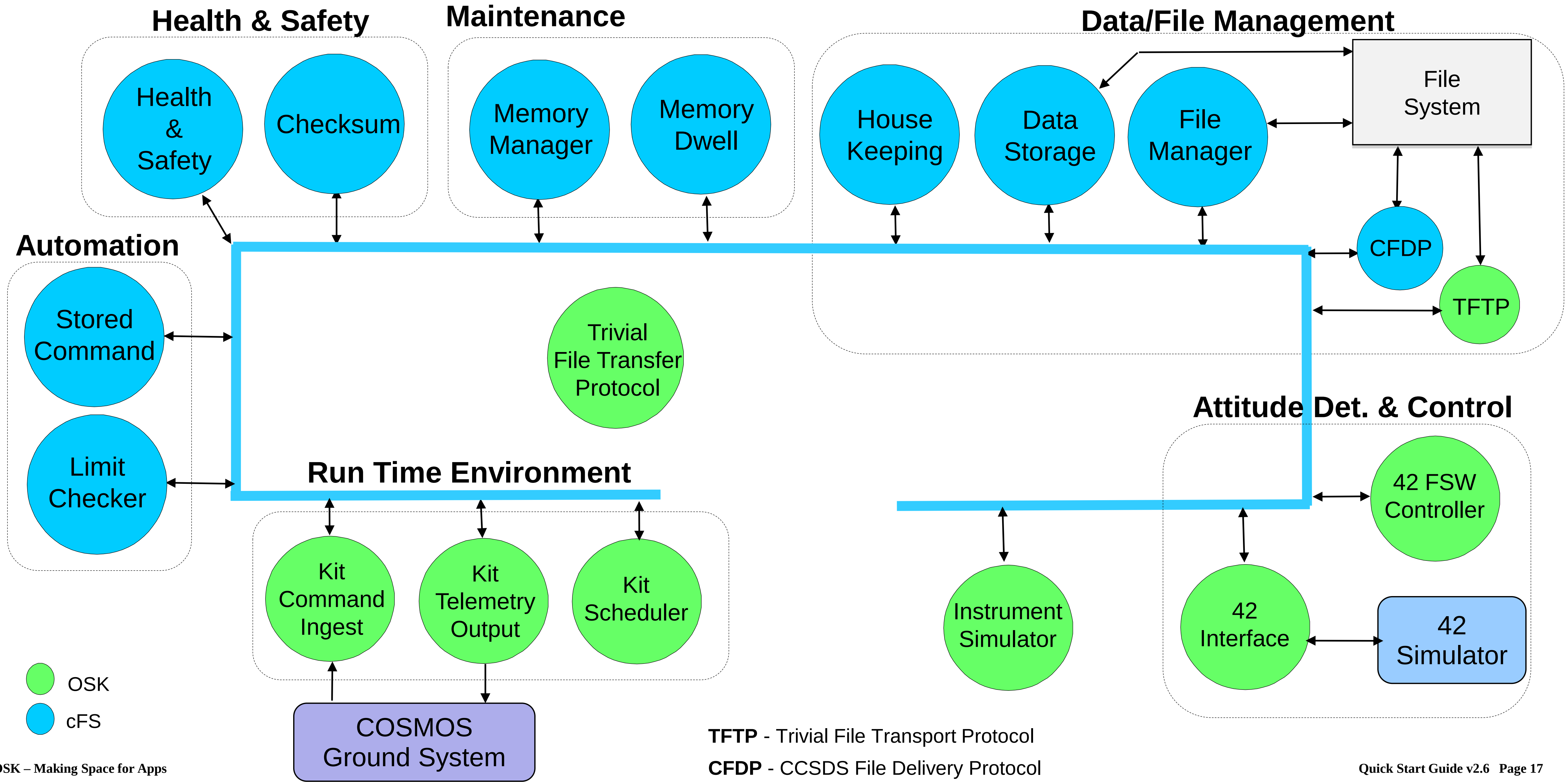
Each service generates a periodic "housekeeping" telemetry packet every few seconds

- The 'Status' section displays a portion of the housekeeping packet
- The entire packet can be displayed using the <Display Tlm> button in the Ground Interface section

Simple Satellite (SimSat)

- **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
    - A 1Hz auxiliary spacecraft data containing time, attitude, orbit data, and instrument status
    - Start science during a ground contact. Can be automated but ops prefers to monitor instrument health.
  - Ground contact resources/schedule are preplanned
    - Implies autonomous operations can be loaded on board using stored commands
  - FSW must autonomously monitor instrument health and power off the instrument in the event of a fault

SimSat Applications (1 of 4)



● OSK
● cFS

TFTP - Trivial File Transport Protocol

CFDP - CCSDS File Delivery Protocol

SimSat Applications (2 of 4)

- **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

SimSat Applications (3 of 4)

- **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

SimSat Applications (4 of 4)

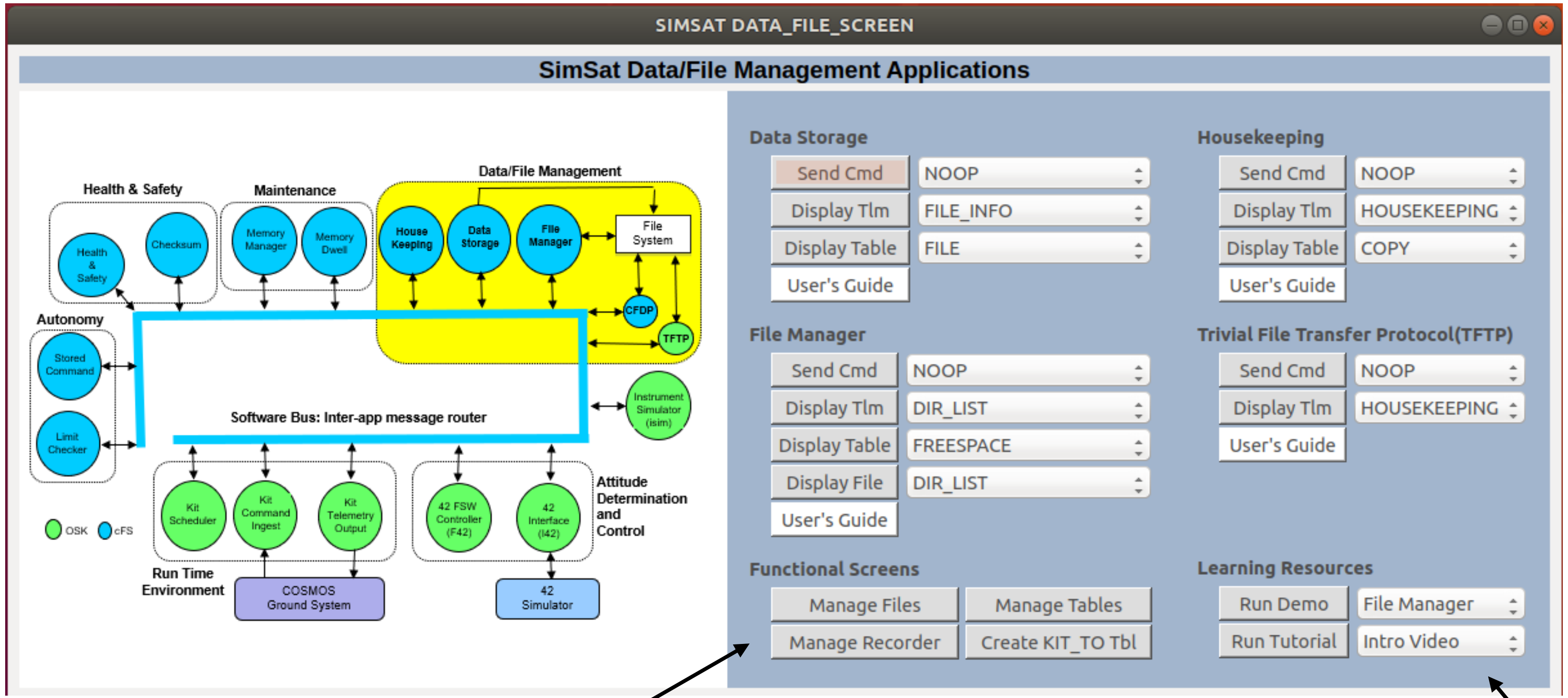
- **Attitude Determination and Control Apps**
  - **42 Interface (I42)** manages a TCP/IP connection to 42 and transfers actuators/sensor packets to/from 42
  - **42 FSW (F42)** Implements the "ThreeAxisFsw" attitude control algorithm defined in 42

- **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

SimSat App Screens

Each functional application group screen uses the following layout



Complete interface to each app

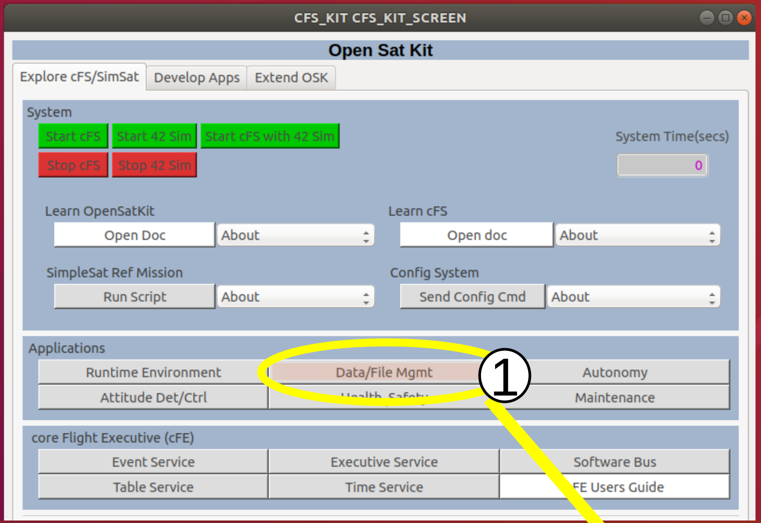
- All commands
- All telemetry packets
- "Display Table" – Dump, transfer and display table in COSMOS Table Manager
- "Display File" – Issue app's command to create a file, then transfer and display binary file in COSMOS Table Manager

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)

App Interactive Resources (1 of 2)

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



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): 0

Learn OpenSatKit		Learn cFS	
Open Doc	About	Open doc	About

SimpleSat Ref Mission		Config System	
Run Script	About	Send Config Cmd	About

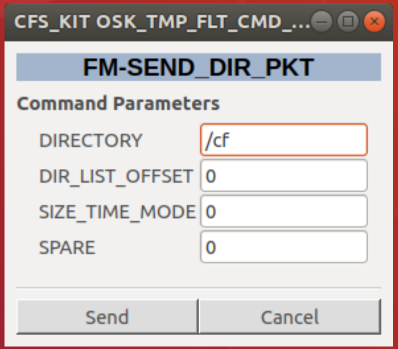
Applications

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

①



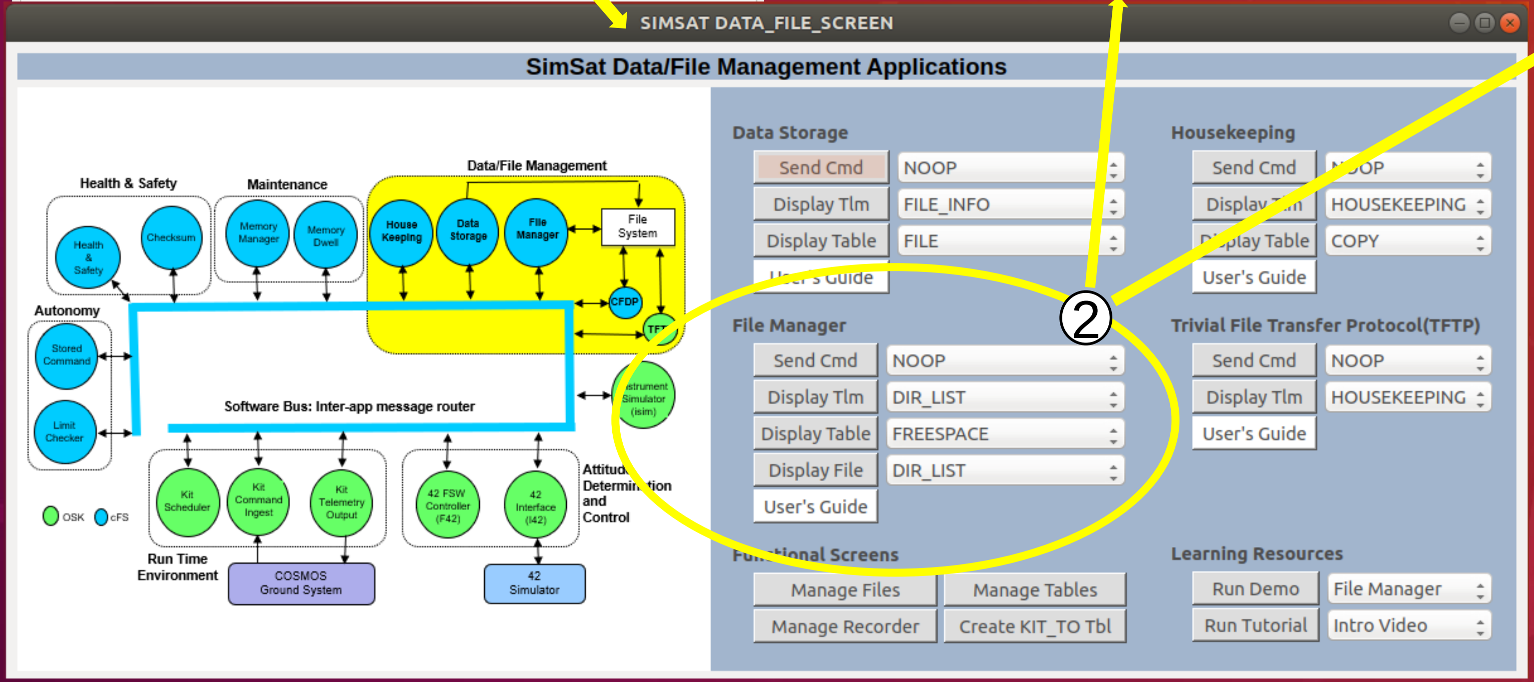
CFS_KIT OSK_TMP_FLT_CMD_...

FM-SEND_DIR_PKT

Command Parameters

DIRECTORY	/cf
DIR_LIST_OFFSET	0
SIZE_TIME_MODE	0
SPARE	0

Send | Cancel



SIMSAT DATA_FILE_SCREEN

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	

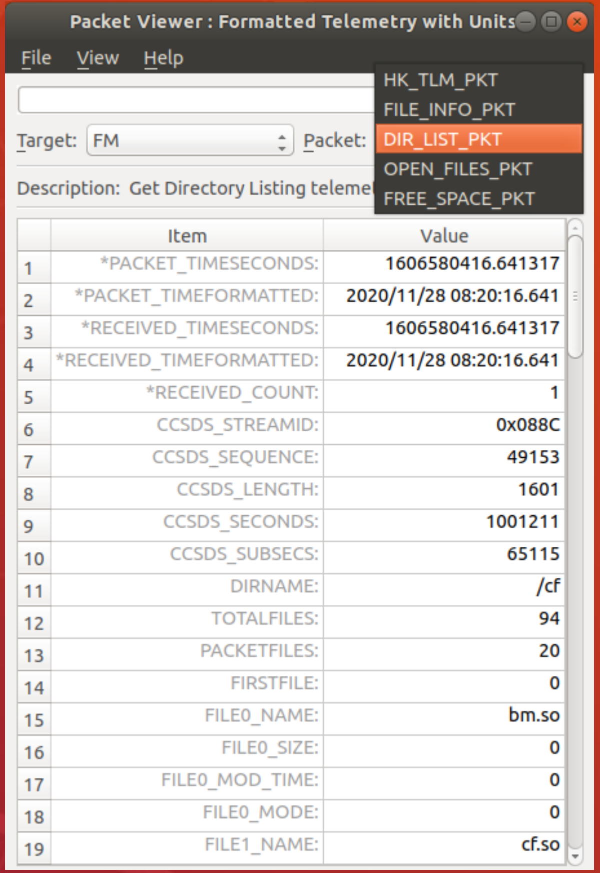
Functional Screens

Manage Files	Manage Tables
Manage Recorder	Create KIT_TO Tbl

Learning Resources

Run Demo	File Manager
Run Tutorial	Intro Video

②



Packet Viewer : Formatted Telemetry with Units

File View Help

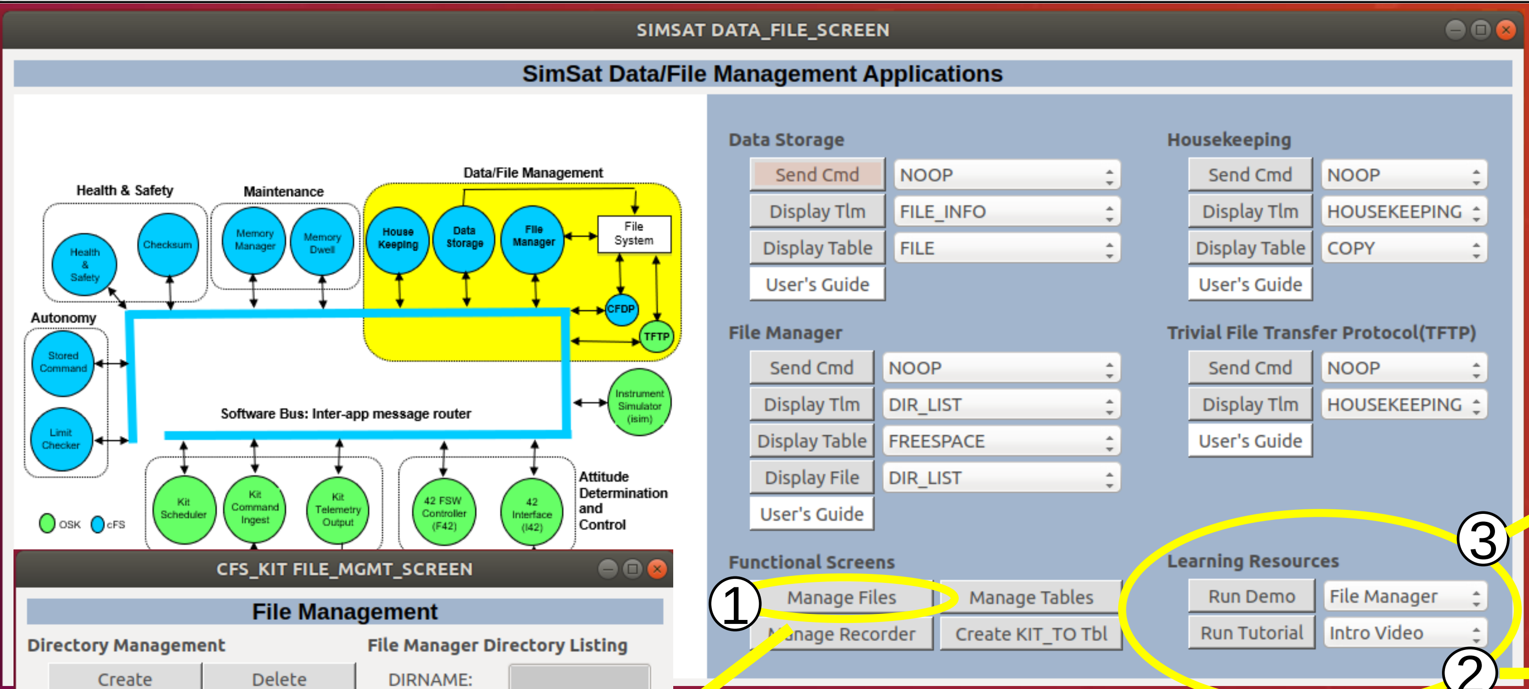
Target: FM Packet: DIR_LIST_PKT

HK_TLM_PKT
FILE_INFO_PKT
DIR_LIST_PKT
OPEN_FILES_PKT
FREE_SPACE_PKT

Description: Get Directory Listing teleme

	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

App Interactive Resources (2 of 2)


SIMSAT DATA_FILE_SCREEN

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	

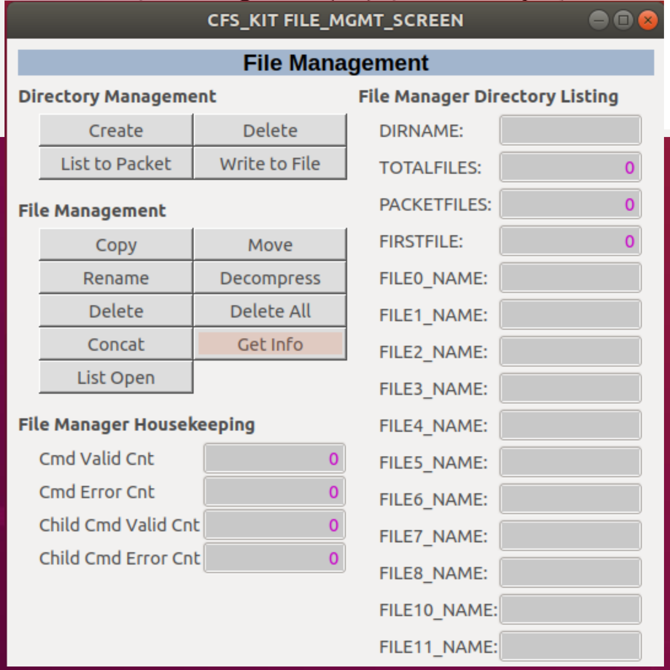
Functional Screens

Manage Files	Manage Tables
Manage Recorder	Create KIT_TO Tbl

Learning Resources

Run Demo	File Manager
Run Tutorial	Intro Video

1 2 3


CFS_KIT FILE_MGMT_SCREEN

File Management

Directory Management

Create	Delete
List to Packet	Write to File

File Management

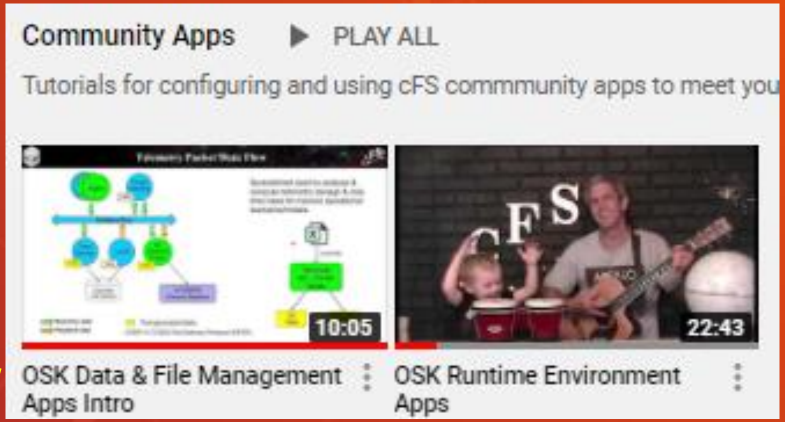
Copy	Move
Rename	Decompress
Delete	Delete All
Concat	Get Info
List Open	

File Manager Housekeeping

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

File Manager Directory Listing

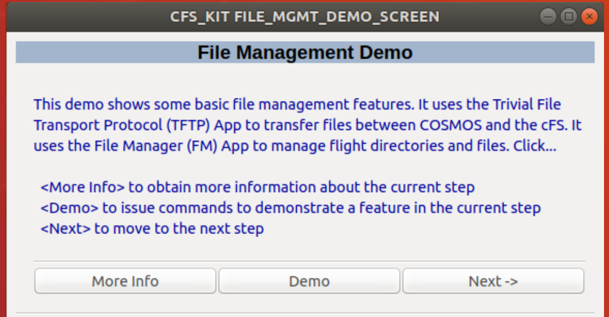
DIRNAME:	
TOTALFILES:	0
PACKETFILES:	0
FIRSTFILE:	0
FILE0_NAME:	
FILE1_NAME:	
FILE2_NAME:	
FILE3_NAME:	
FILE4_NAME:	
FILE5_NAME:	
FILE6_NAME:	
FILE7_NAME:	
FILE8_NAME:	
FILE10_NAME:	
FILE11_NAME:	


Community Apps ▶ PLAY ALL

Tutorials for configuring and using cFS commmunity apps to meet you

OSK Data & File Management Apps Intro

OSK Runtime Environment Apps


CFS_KIT FILE_MGMT_DEMO_SCREEN

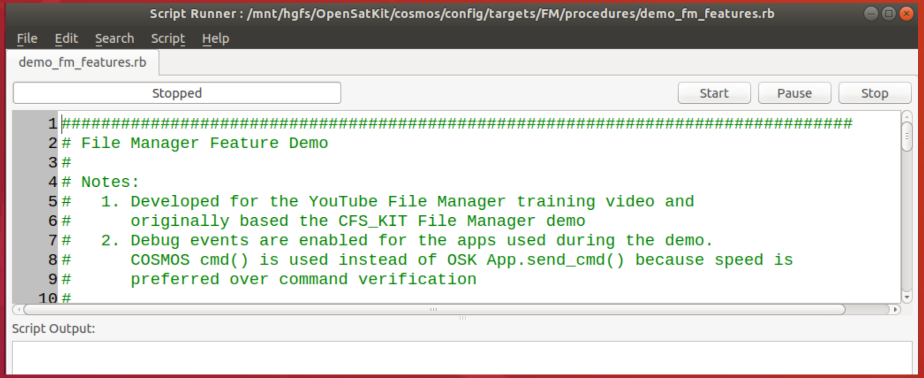
File Management Demo

This demo shows some basic file management features. It uses the Trivial File Transfer 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 ->

1. FM summary page with HK and directory listing
2. FM feature demo
3. YouTube Tutorials


Script Runner : /mnt/hgfs/OpenSatKit/cosmos/config/targets/FM/procedures/demo_fm_features.rb

File Edit Search Script Help

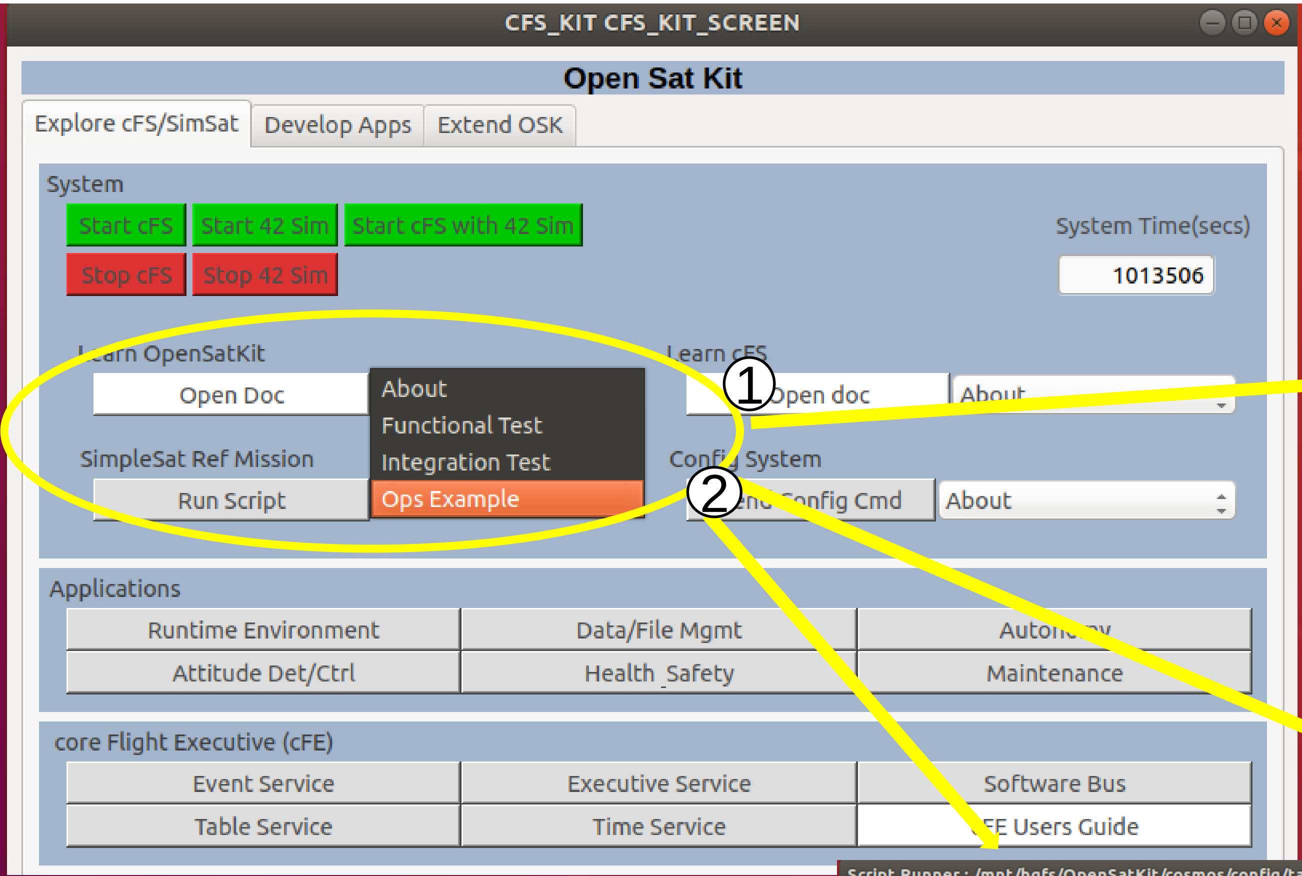
demo_fm_features.rb

Stopped | Start | Pause | Stop

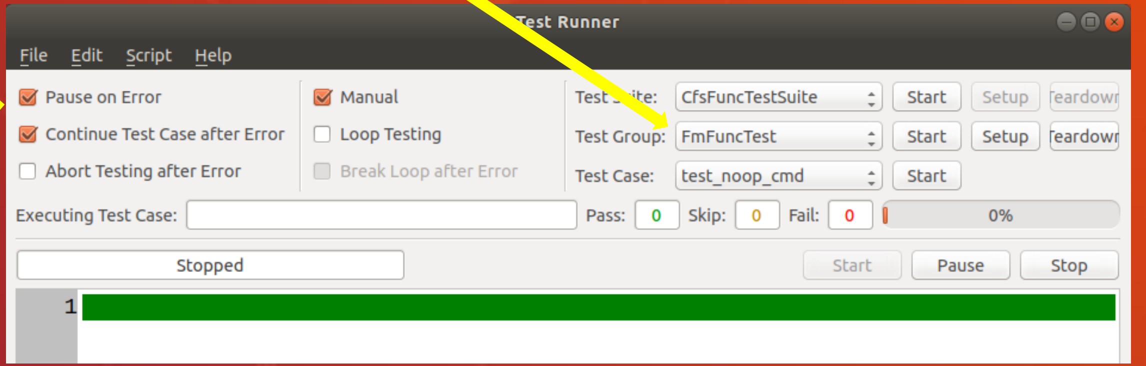
```
################################################################################
# File Manager Feature Demo
#
# Notes:
#   1. Developed for the YouTube File Manager training video and
#      originally based the CFS_KIT File Manager demo
#   2. Debug events are enabled for the apps used during the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

Script Output:

App System Scripting Resources

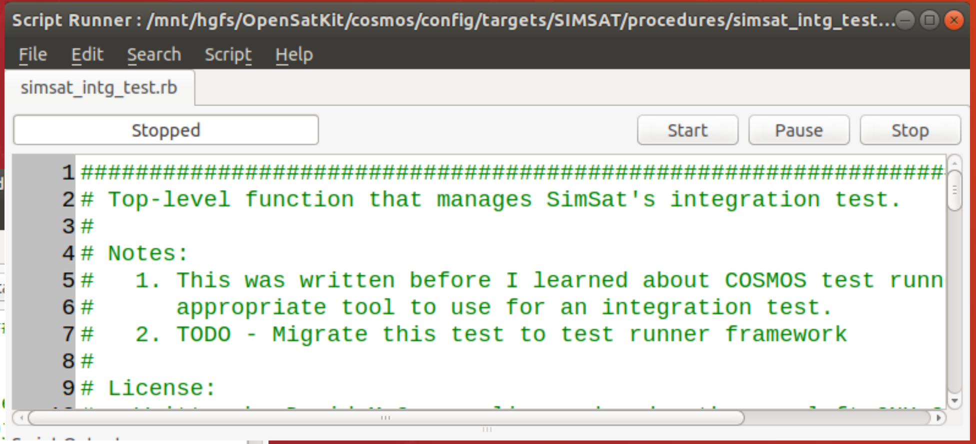


1. Functional Test Suites contain Test Groups for each app that run in the COSMOS Test Runner



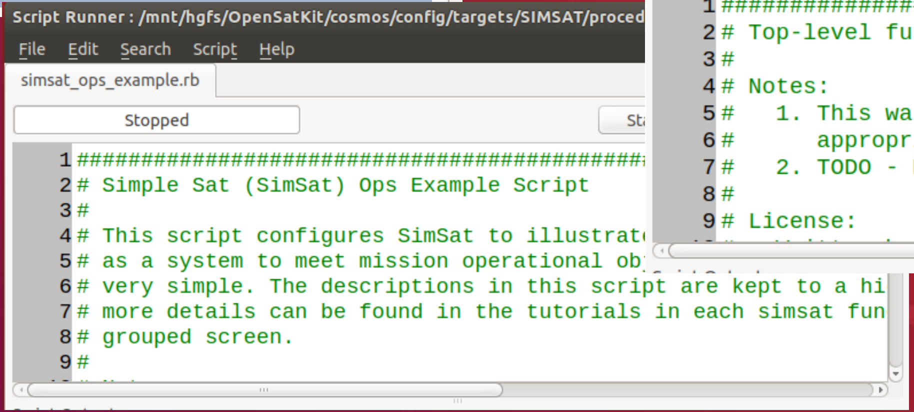
- [✓] Pause on Error
- [✓] Continue Test Case after Error
- [ ] Abort Testing after Error
- [✓] Manual
- [ ] Loop Testing
- [ ] Break Loop after Error

Test Suite: CfsFuncTestSuite
Test Group: FmFuncTest
Test Case: test_noop_cmd
Executing Test Case:
Pass: 0 Skip: 0 Fail: 0 0%
Stopped



Script Runner : /mnt/hgfs/OpenSatKit/cosmos/config/targets/SIMSAT/procedures/simsat_intg_test...
File
Edit
Search
Script
Help
simsat_intg_test.rb
Stopped
Start
Pause
Stop

```
##################################################################
# Top-level function that manages SimSat's integration test.
#
# Notes:
#   1. This was written before I learned about COSMOS test runn
#      appropriate tool to use for an integration test.
#   2. TODO - Migrate this test to test runner framework
#
# License:
```



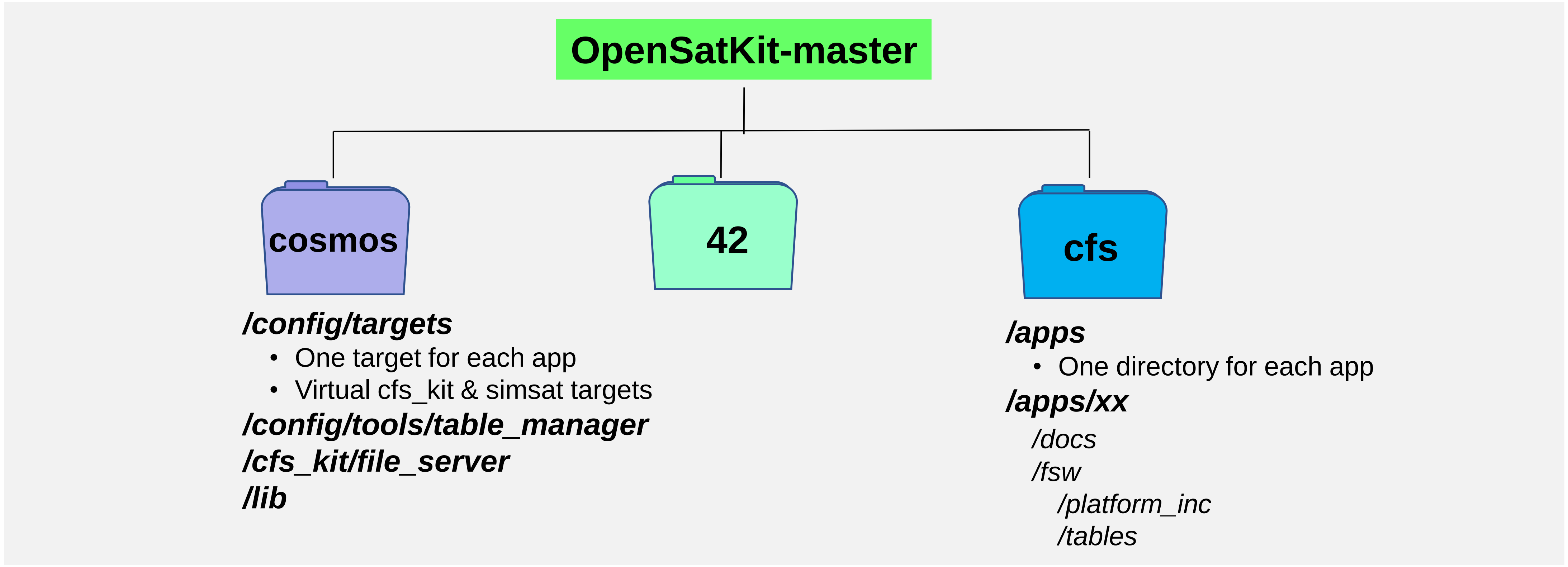
Script Runner : /mnt/hgfs/OpenSatKit/cosmos/config/targets/SIMSAT/proced
File
Edit
Search
Script
Help
simsat_ops_example.rb
Stopped
Sta

```
################################################
# Simple Sat (SimSat) Ops Example Script
#
# This script configures SimSat to illustrate
# as a system to meet mission operational ob
# very simple. The descriptions in this script are kept to a hi
# more details can be found in the tutorials in each simsat fun
# grouped screen.
#
```

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

OSK System Overview

Directory Highlights



- **COSMOS Targets are architectural components that define remote systems that communicate through COSMOS Interfaces**
  - Contain command, telemetry and screen definitions and ruby procedure & library scripts
  - ***cfs_kit*** target defines OSK screens and ruby scripts that have an OSK scope
  - ***simsat*** target defines screens and ruby scripts that are specific to the SimSat reference mission
- **/cosmos/config/tools/table_manager** contains binary file and table definition files
- **/cosmos/cfs_kit/file_server** used for transferring files between ground and flight. “tables” subdirectory used for table transfers
- **/cosmos/lib** defines OSK extensions to COSMOS

COSMOS Configuration (1 of 2)

- **COSMOS *Target* (OpenSatKit/cosmos/config/targets)**
  - Architectural component, typically on an embedded system, that COSMOS can send commands to and receive telemetry from
  - For each target users can define command packets, telemetry packets, screens, and Ruby scripts.
  - Each FSW application is defined as a target
  - OSK defines a virtual target ***CFS_KIT*** to serve as the User's primary interface
  - OSK defines a virtual target ***SIMSAT*** to serve as a reference mission

- **OSK scripts in *OpenSatKit/cosmos/lib* extend COSMOS scripting API**
  - API documentation is under development. See code for details

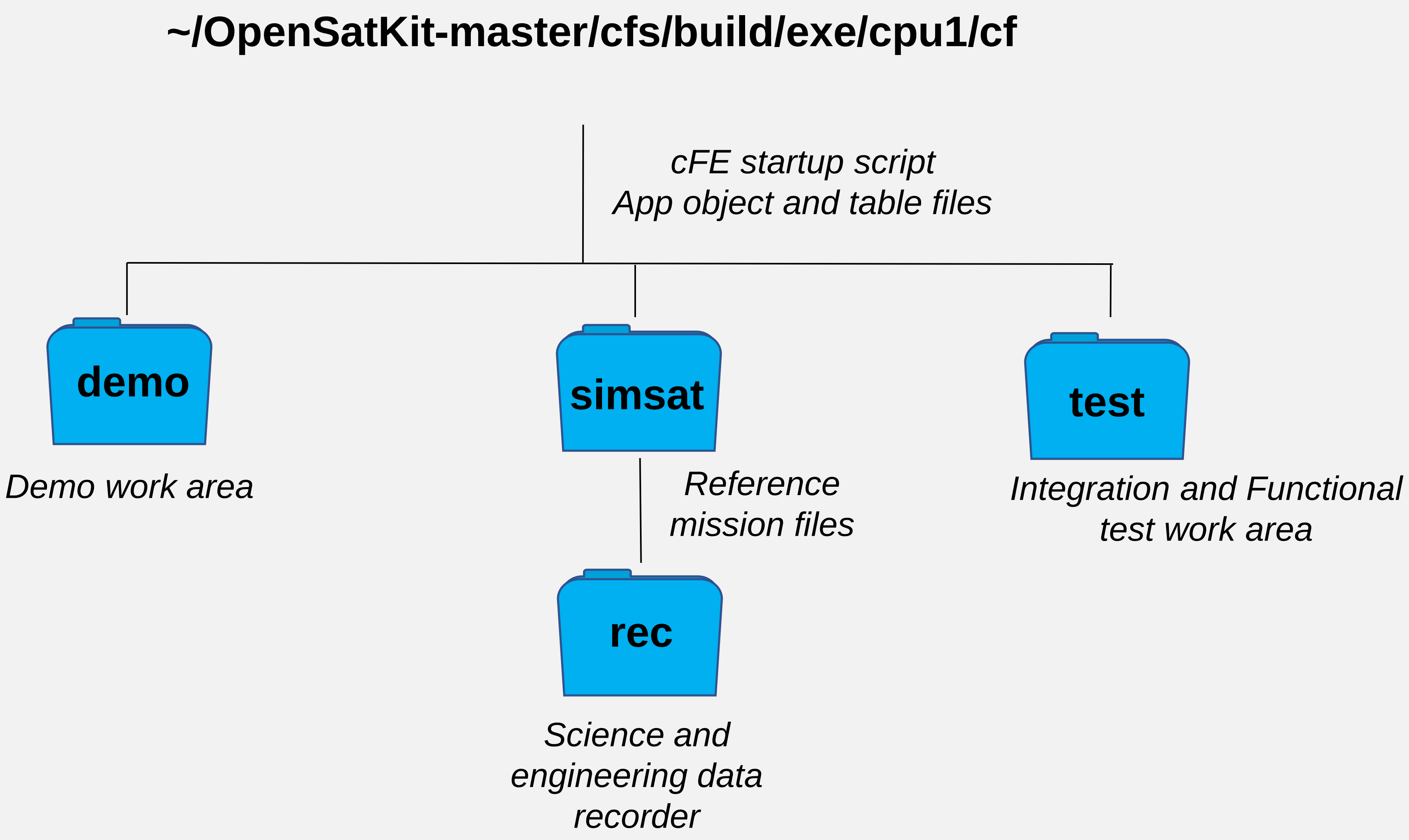
COSMOS Configuration (2 of 2)

- **OSK specific directories defined in *OpenSatKit/cosmos/cfs_kit***
  - */docs*: cFE and OSK documentation
  - */file_server*: Default location for file transferred to/from FSW
    - */table* subdirectory contains table files
    - COSMOS Table Manager file formats defined in */cosmos/config/tools/TableManager*
  - */tools*: cFE and OSK standalone tools
  - */tutorials*: Tutorial files

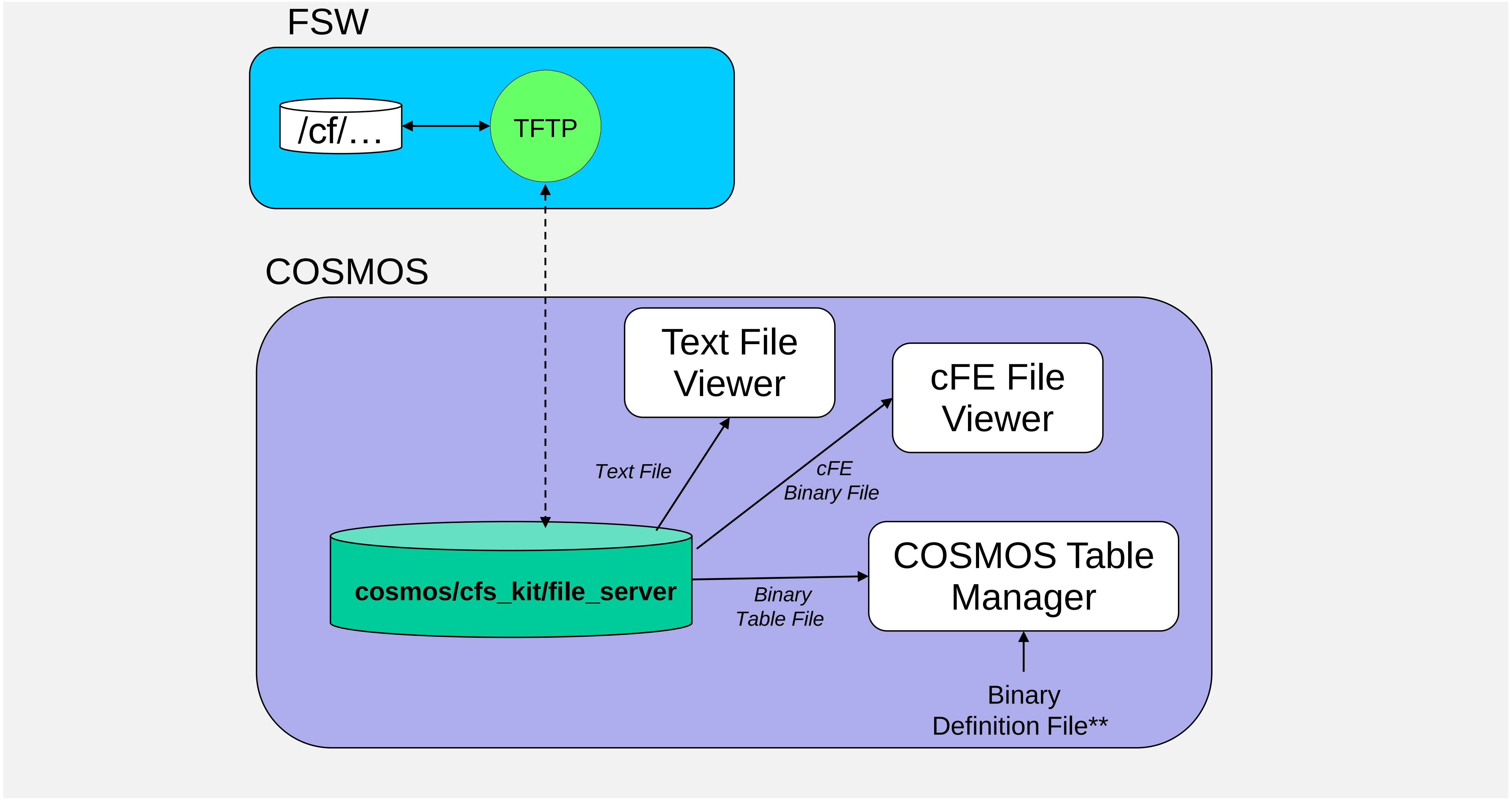
OSK Conventions

- **Most cFE services have commands that can generate a telemetry as part of the response or write information to a file**
  - The verbs *list* and *send* indicate information is sent in a telemetry packet.
  - *Write* is used when information is written to a file
- **The FSW directory /cf (compact flash) is used as the default location for onboard file creation and flight-ground file transfers**
  - This is mapped to *OpenSatKit/cfs/build/exe/cpu1/cf*
- **OpenSatKit/cosmos/cfs_kit/file_server is used as the default ground file location**
  - Table are located in the *tables* subdirectory
- **OSK often uses osk_tmp_bin.dat as a standard temporary binary file name to avoid clutter**
- **OSK does not "cheat" when working with ground and flight tables**
  - **Files are transferred between flight and ground locations and not accessed via shared locations within the VM**

Flight Software Directories


~/OpenSatKit-master/cfs/build/exe/cpu1/cf
cFE startup script
App object and table files
demo
Demo work area
simsat
Reference mission files
test
Integration and Functional test work area
rec
Science and engineering data recorder

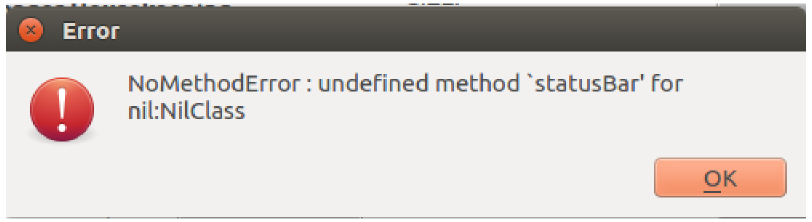
File Management



** Definition files in ~/cosmos/config/tools/table_manager

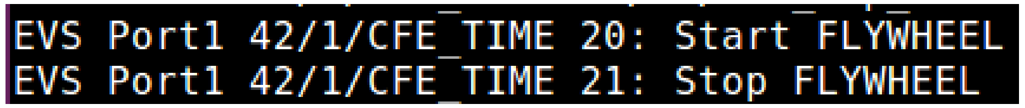
Minor Inconveniences (1 of 2)

- **OSK is a work in progress with a few known issues that you can ignore**
- **If you cancel an OSK dialogue you may see the follow COSMOS error dialogue.**



```
NoMethodError : undefined method `statusBar' for
nil:NilClass
```

- **The FSW terminal window may display start and stop "FlyWheel" messages**
  - OSK is a non-realtime environment so the cFE time service is warning that's it's not operating within its real-time precision limits relative to a 1Hz timer
  - OSK is designed to help users learn functional features and only requires reasonable timing performance in order for the scheduler to execute its schedule correctly



```
EVS Port1 42/1/CFE_TIME 20: Start FLYWHEEL
EVS Port1 42/1/CFE_TIME 21: Stop FLYWHEEL
```

Minor Inconveniences (2 of 2)

- Some cFS binary files are variable length. The Table Manager definition files support fixed length files, therefore you may see an error dialog stating the file doesn't contain all of the records. This message is from cFE Executive Service Task Information file.

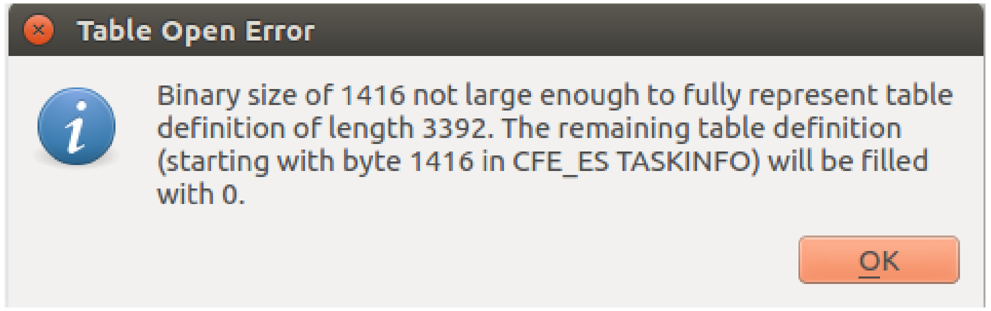


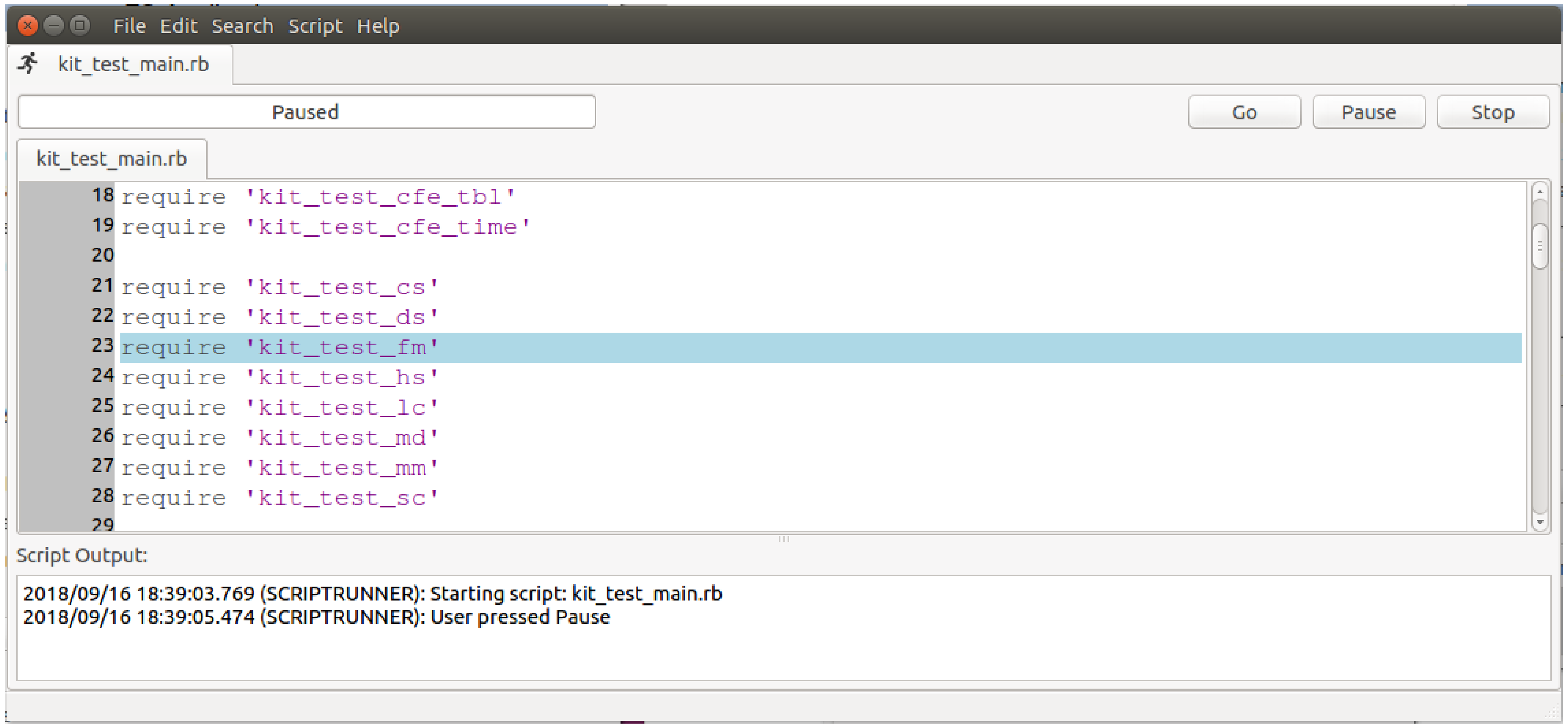
Table Open Error

Binary size of 1416 not large enough to fully represent table definition of length 3392. The remaining table definition (starting with byte 1416 in CFE_ES TASKINFO) will be filled with 0.

OSK Scripting Overview

UNDER CONSTRUCTION

SimSat Integration Script



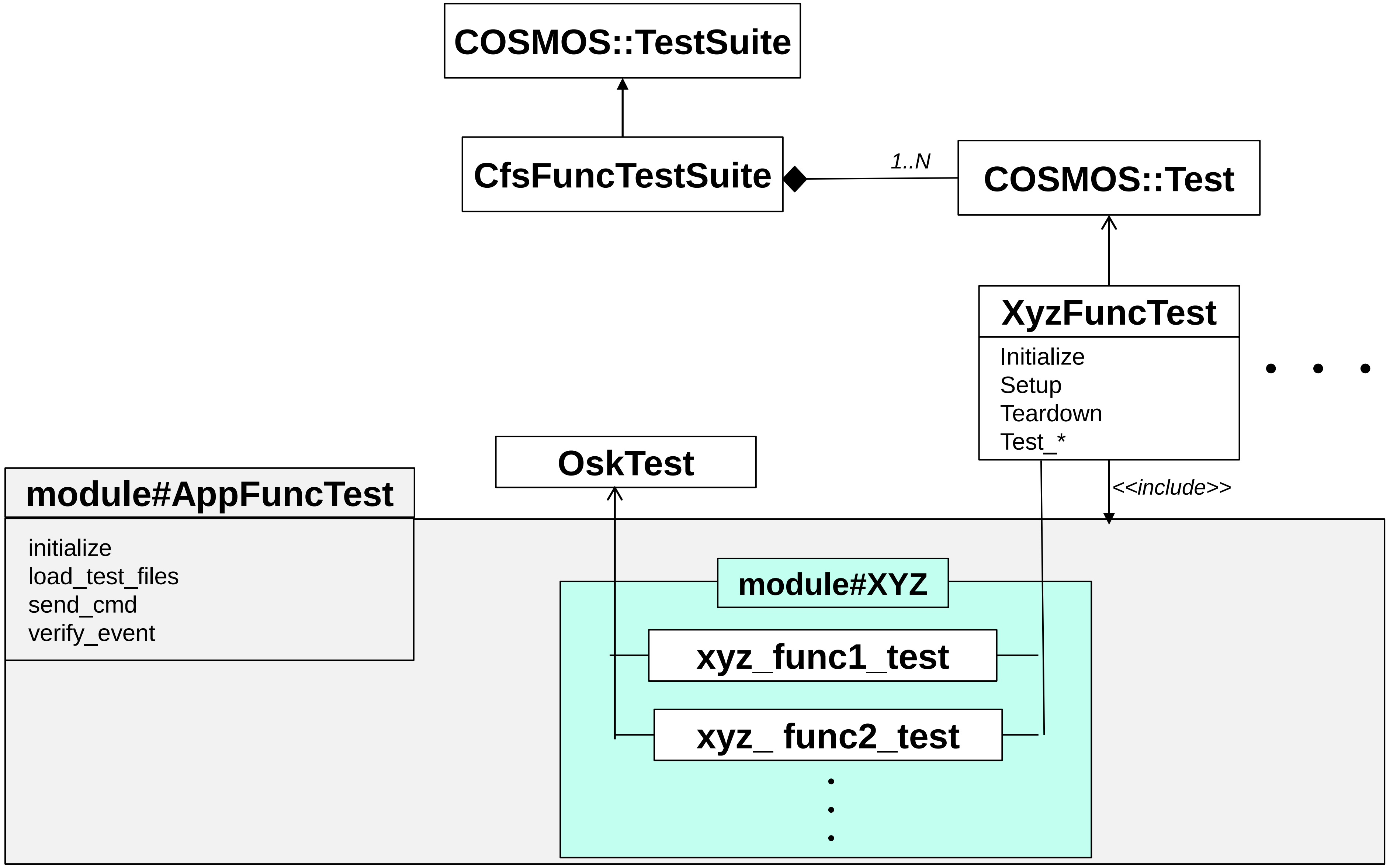
```
require 'kit_test_cfe_tbl'
require 'kit_test_cfe_time'

require 'kit_test_cs'
require 'kit_test_ds'
require 'kit_test_fm'
require 'kit_test_hs'
require 'kit_test_lc'
require 'kit_test_md'
require 'kit_test_mm'
require 'kit_test_sc'
```

```
2018/09/16 18:39:03.769 (SCRIPTRUNNER): Starting script: kit_test_main.rb
2018/09/16 18:39:05.474 (SCRIPTRUNNER): User pressed Pause
```

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

Functional Test Scripts

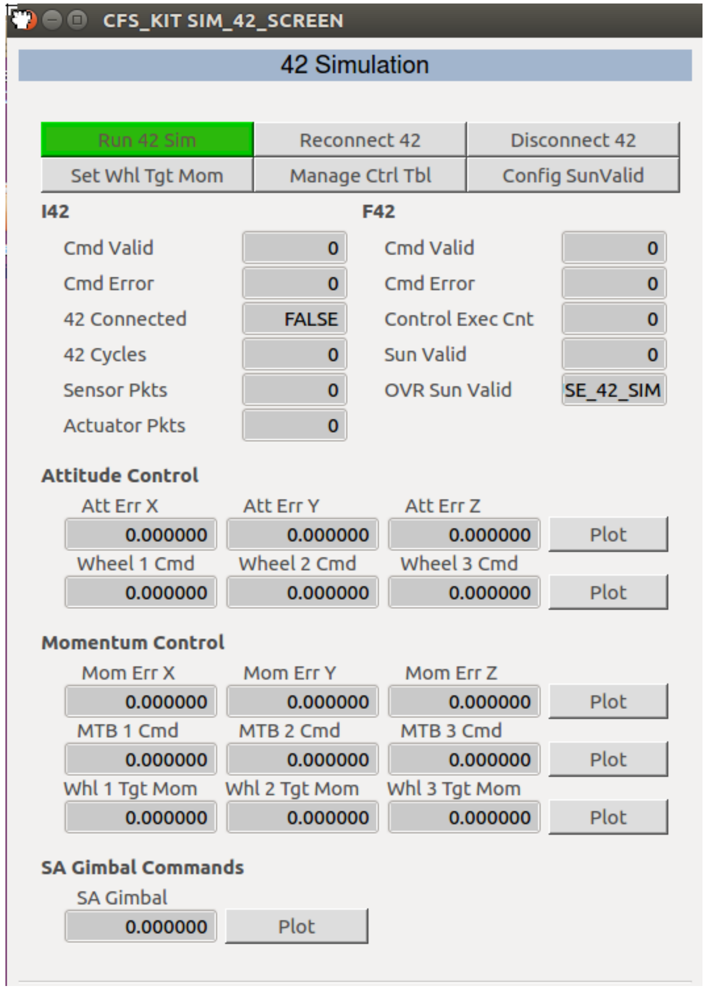


SimSat Operation Script Example

Running with 42 Simulator

Needs updates since v2.4

Starting the Simulation


CFS_KIT SIM_42_SCREEN

42 Simulation

I42		F42	
Cmd Valid	0	Cmd Valid	0
Cmd Error	0	Cmd Error	0
42 Connected	FALSE	Control Exec Cnt	0
42 Cycles	0	Sun Valid	0
Sensor Pkts	0	OVR Sun Valid	SE_42_SIM
Actuator Pkts	0		

Attitude Control

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

Momentum Control

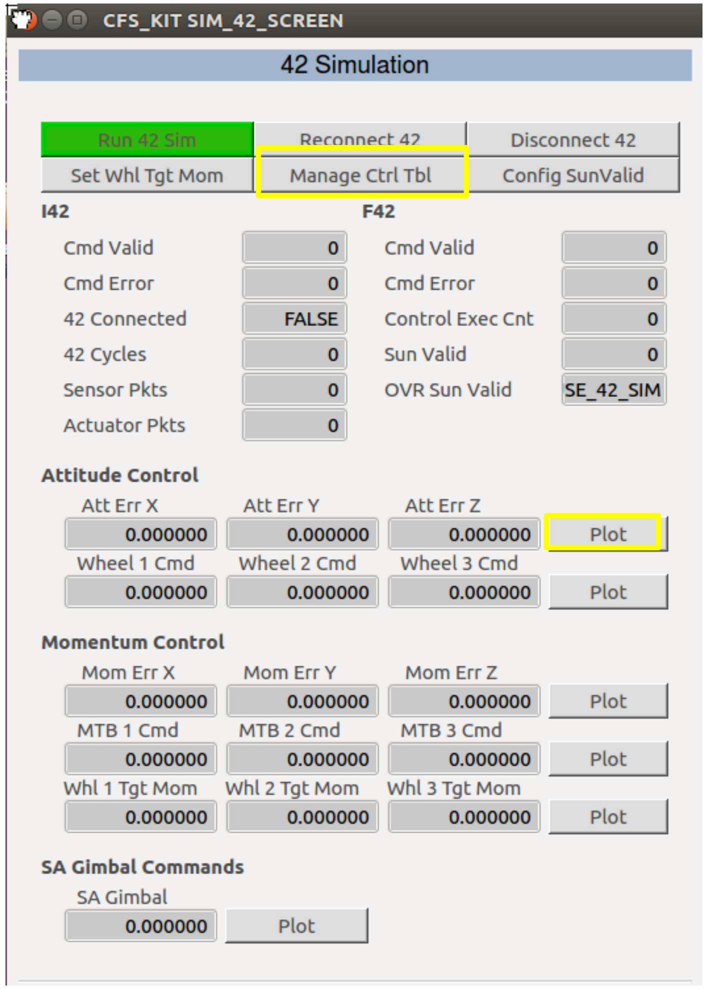
Mom Err X	Mom Err Y	Mom Err Z	
0.000000	0.000000	0.000000	Plot
MTB 1 Cmd	MTB 2 Cmd	MTB 3 Cmd	
0.000000	0.000000	0.000000	Plot
Whl 1 Tgt Mom	Whl 2 Tgt Mom	Whl 3 Tgt Mom	
0.000000	0.000000	0.000000	Plot

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

Preparing 42 Simulation



I42		F42	
Cmd Valid	0	Cmd Valid	0
Cmd Error	0	Cmd Error	0
42 Connected	FALSE	Control Exec Cnt	0
42 Cycles	0	Sun Valid	0
Sensor Pkts	0	OVR Sun Valid	SE_42_SIM
Actuator Pkts	0		

Attitude Control

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

Momentum Control

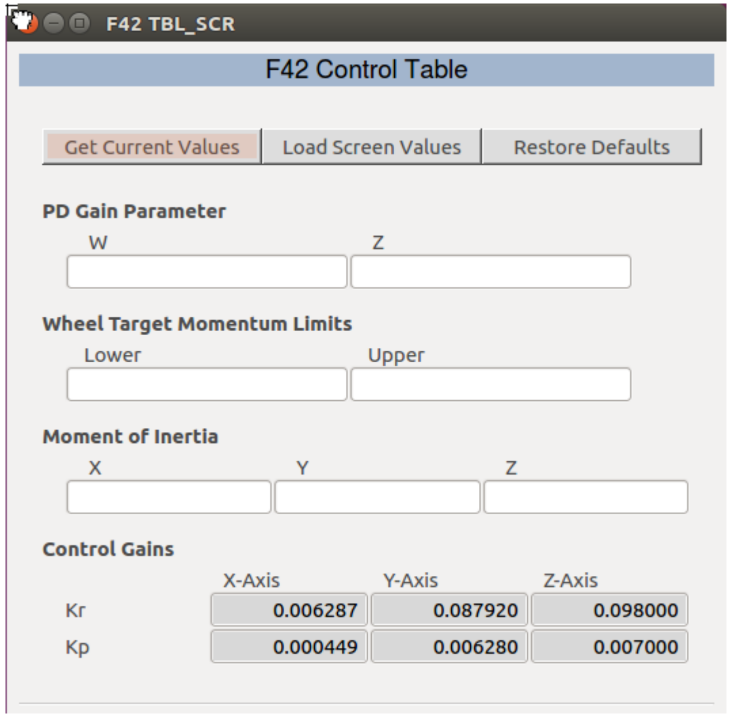
Mom Err X	Mom Err Y	Mom Err Z	
0.000000	0.000000	0.000000	Plot
MTB 1 Cmd	**MTB 2 Cmd**	**MTB 3 Cmd**	
0.000000	0.000000	0.000000	Plot
Whl 1 Tgt Mom	**Whl 2 Tgt Mom**	**Whl 3 Tgt Mom**	
0.000000	0.000000	0.000000	Plot

SA Gimbal Commands

SA Gimbal	
0.000000	Plot

- **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

Managing Control Table



F42 TBL_SCR

F42 Control Table

PD Gain Parameter

W	Z

Wheel Target Momentum Limits

Lower	Upper

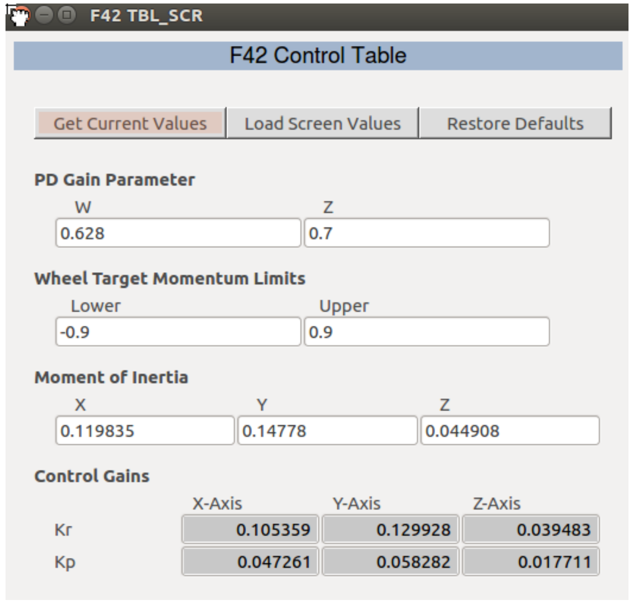
Moment of Inertia

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

PD Gain Parameter

W	Z
0.628	0.7

Wheel Target Momentum Limits

Lower	Upper
-0.9	0.9

Moment of Inertia

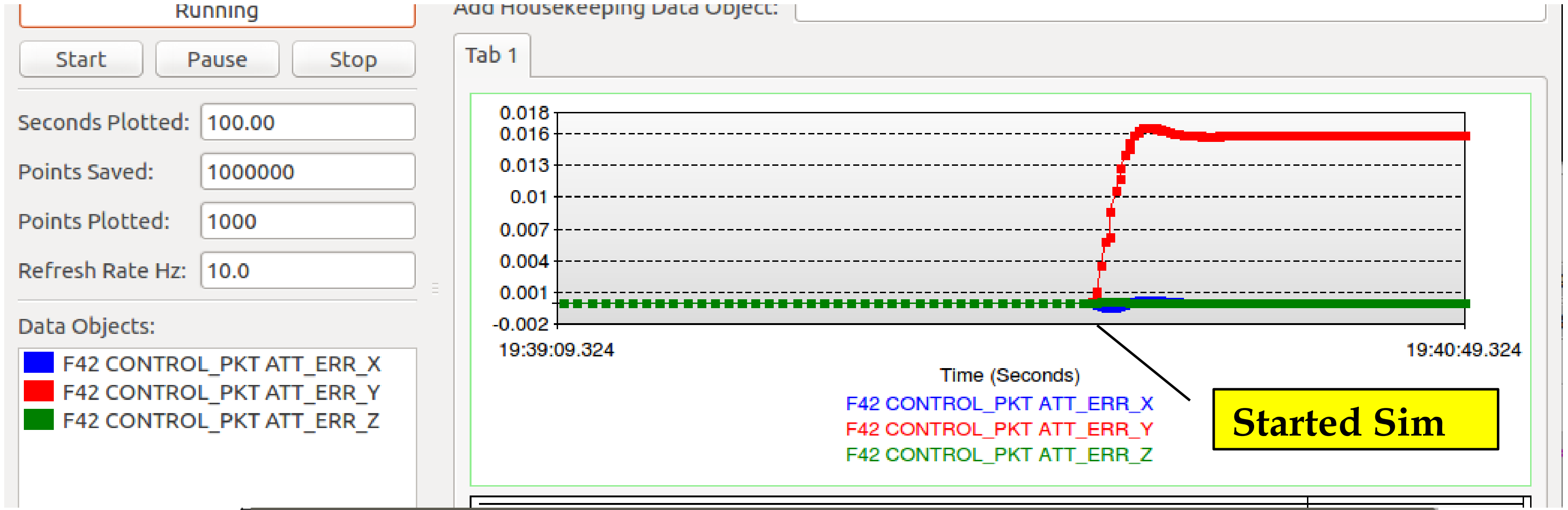
X	Y	Z
0.119835	0.14778	0.044908

Control Gains

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

Plot Attitude Errors

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



Running

Add Housekeeping Data Object:

Seconds Plotted: 100.00

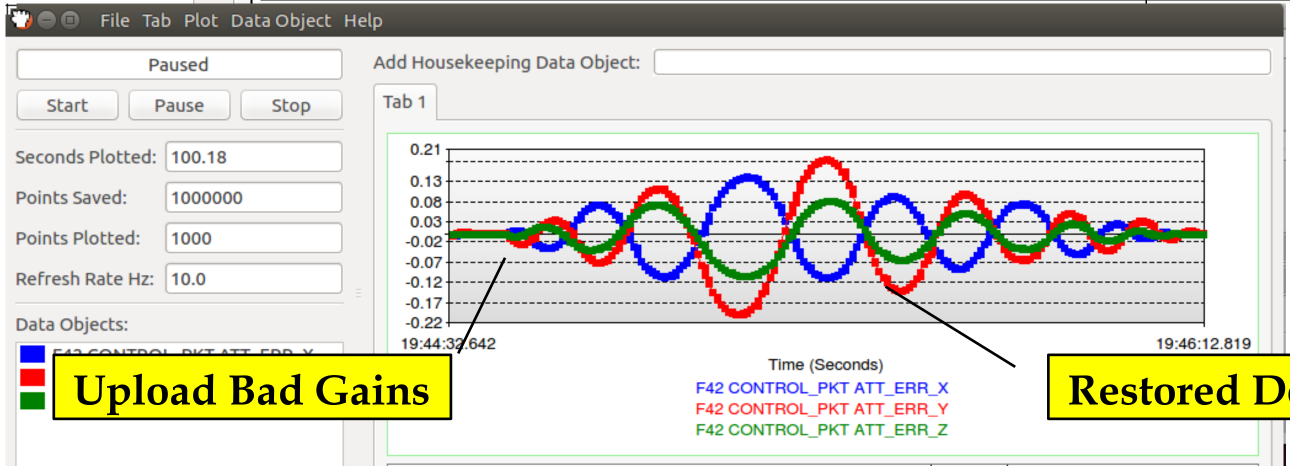
Points Saved: 1000000

Points Plotted: 1000

Refresh Rate Hz: 10.0

Data Objects:

🟦 F42 CONTROL_PKT ATT_ERR_X
🟥 F42 CONTROL_PKT ATT_ERR_Y
🟩 F42 CONTROL_PKT ATT_ERR_Z



Paused

Add Housekeeping Data Object:

Seconds Plotted: 100.18

Points Saved: 1000000

Points Plotted: 1000

Refresh Rate Hz: 10.0

Data Objects:

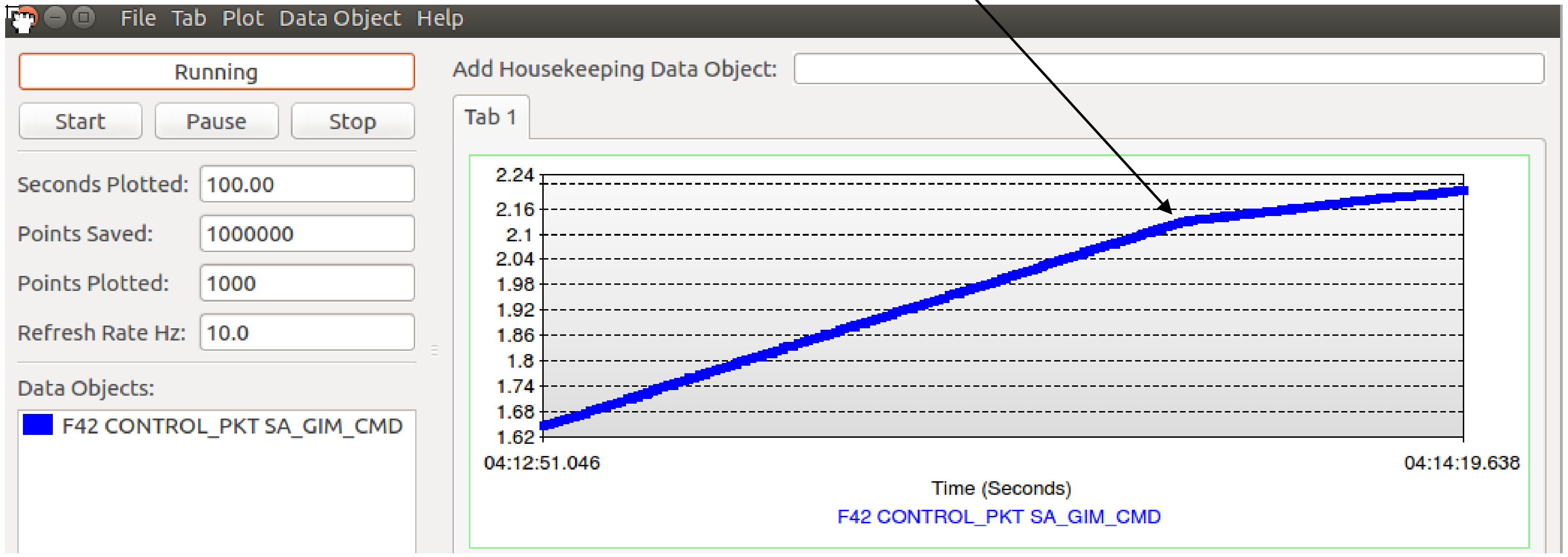
🟦
🟥
🟩

Additional Configuration Options

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

Configure Sun Valid

- 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.



Add Housekeeping Data Object:

Seconds Plotted: 100.00

Points Saved: 1000000

Points Plotted: 1000

Refresh Rate Hz: 10.0

Data Objects:

🟦 F42 CONTROL_PKT SA_GIM_CMD

Sim Termination

1. Click ***<Disconnect 42>*** to end a 42 simulation that is running with the FSW

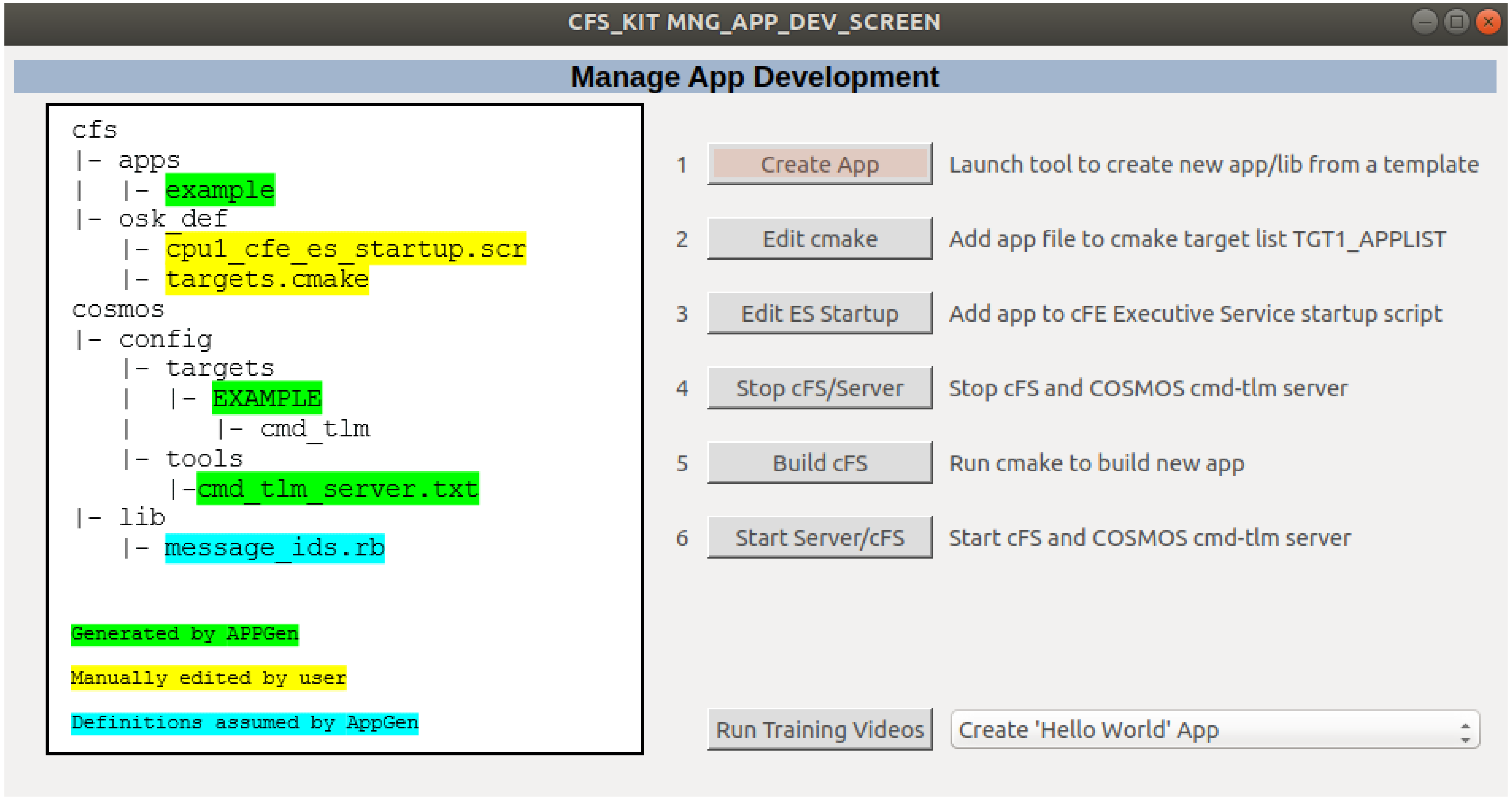
2. To terminate the flight software click on the terminal window with the FSW messages and then enter ctrl-c

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.

Managing Apps

UNDER CONSTRUCTION

Create App



CFS_KIT MNG_APP_DEV_SCREEN

Manage App Development

```
cfs
|- apps
|   |- example
|- osk_def
    |- cpu1_cfe_es_startup.scr
    |- targets.cmake
cosmos
|- config
    |- targets
    |   |- EXAMPLE
    |       |- cmd_tlm
    |- tools
        |-cmd_tlm_server.txt
|- lib
    |- message_ids.rb
```

Generated by APPGen

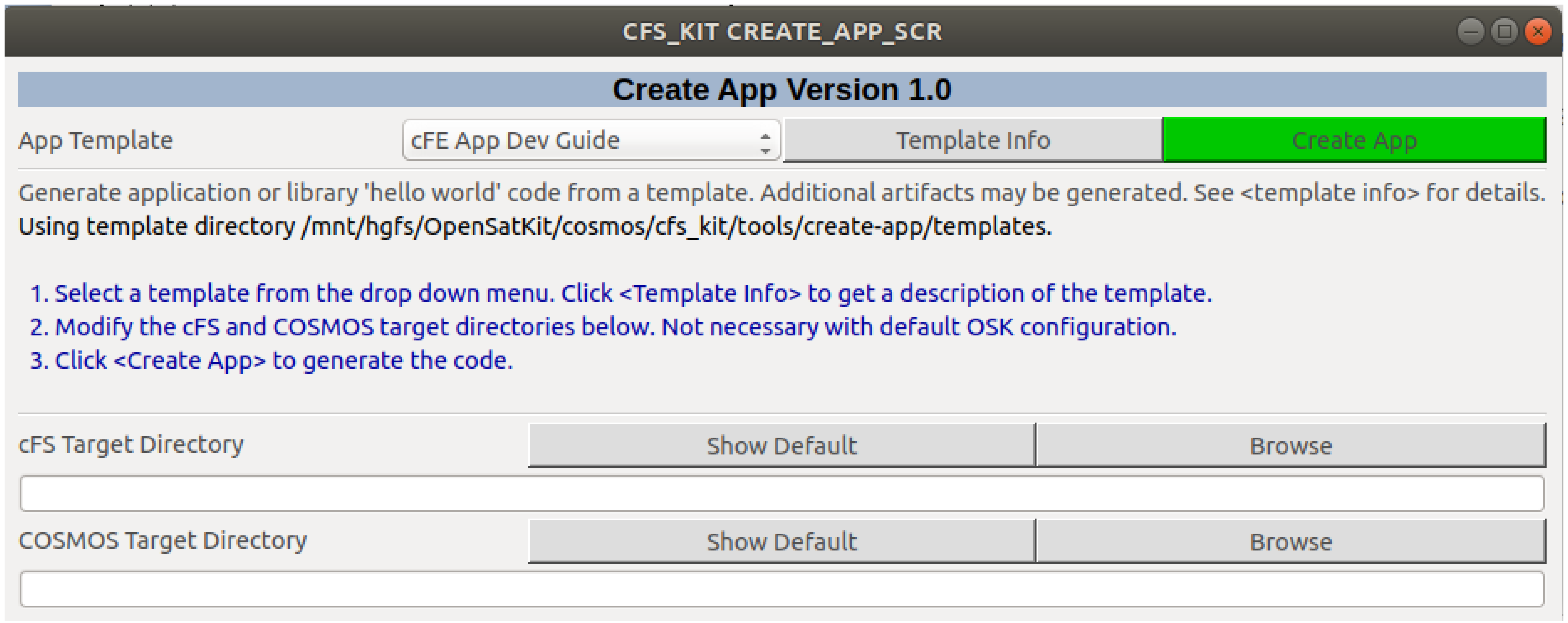
Manually edited by user

Definitions assumed by AppGen

1	Create App	Launch tool to create new app/lib from a template
2	Edit cmake	Add app file to cmake target list TGT1_APPLIST
3	Edit ES Startup	Add app to cFE Executive Service startup script
4	Stop cFS/Server	Stop cFS and COSMOS cmd-tlm server
5	Build cFS	Run cmake to build new app
6	Start Server/cFS	Start cFS and COSMOS cmd-tlm server

Six quick steps to create a “Hello World” created and integrate it into the kit

Create App Templates



Create App Version 1.0

App Template: cFE App Dev Guide | Template Info | Create App

Generate application or library 'hello world' code from a template. Additional artifacts may be generated. See <template info> for details.
Using template directory /mnt/hgfs/OpenSatKit/cosmos/cfs_kit/tools/create-app/templates.

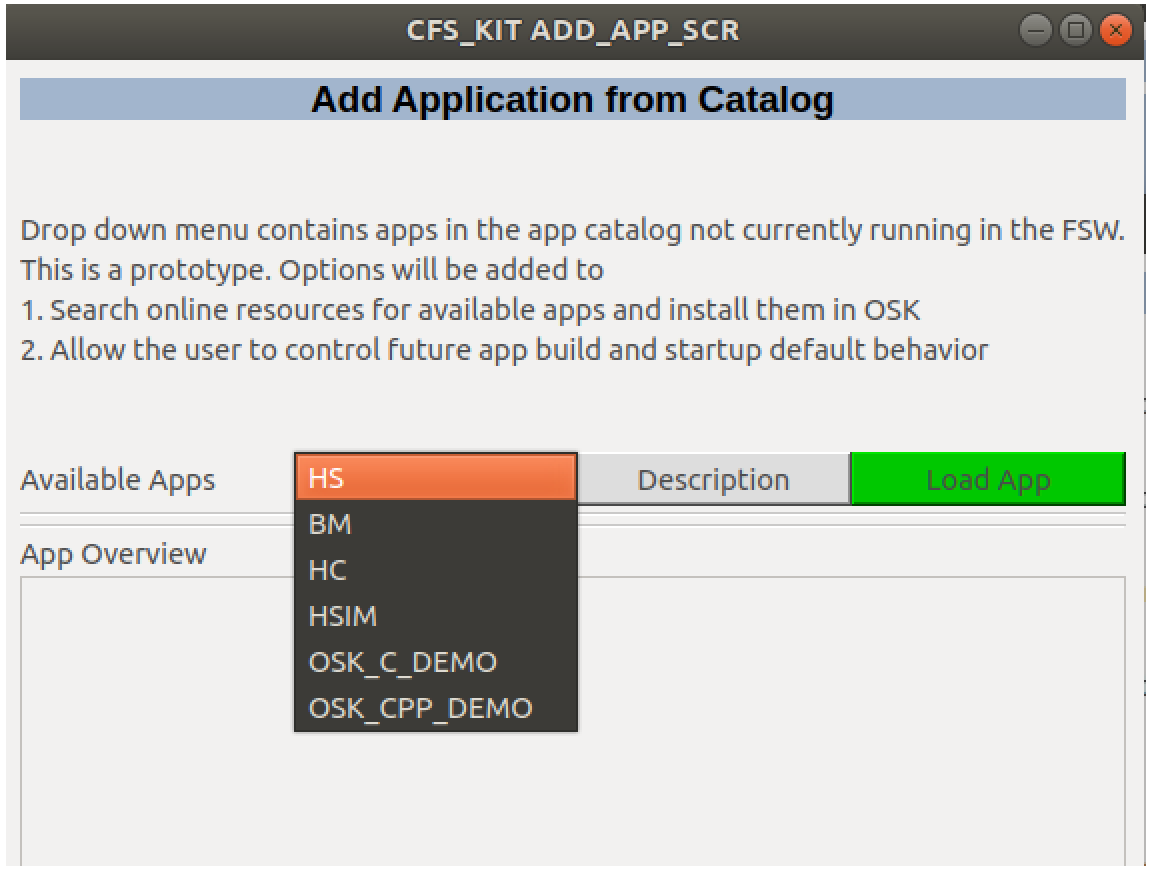
1. Select a template from the drop down menu. Click <Template Info> to get a description of the template.
2. Modify the cFS and COSMOS target directories below. Not necessary with default OSK configuration.
3. Click <Create App> to generate the code.

cFS Target Directory | Show Default | Browse

COSMOS Target Directory | Show Default | Browse

Add App

Early stage prototype for exploring app packaging and distribution concepts

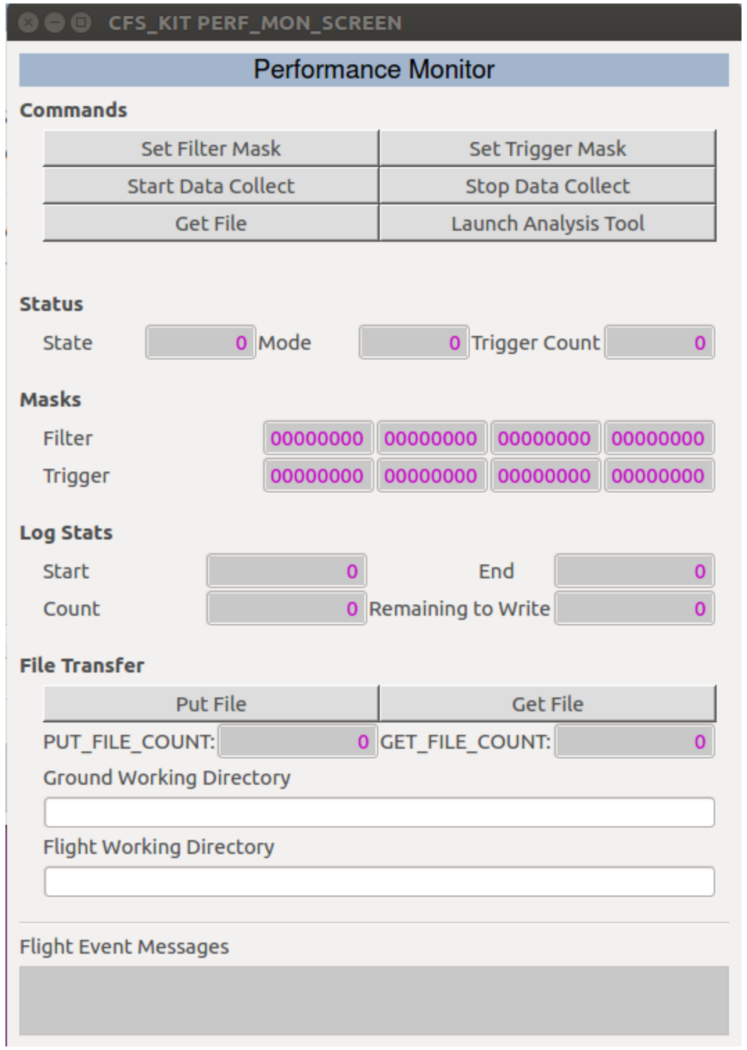


CFS_KIT ADD_APP_SCR

Add Application from Catalog

Drop down menu contains apps in the app catalog not currently running in the FSW.
This is a prototype. Options will be added to
1. Search online resources for available apps and install them in OSK
2. Allow the user to control future app build and startup default behavior

Performance Monitor Tool



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 0 Mode 0 Trigger Count 0

Masks

Filter	00000000	00000000	00000000	00000000
Trigger	00000000	00000000	00000000	00000000

Log Stats

Start 0 End 0

Count 0 Remaining to Write 0

File Transfer

Put File	Get File

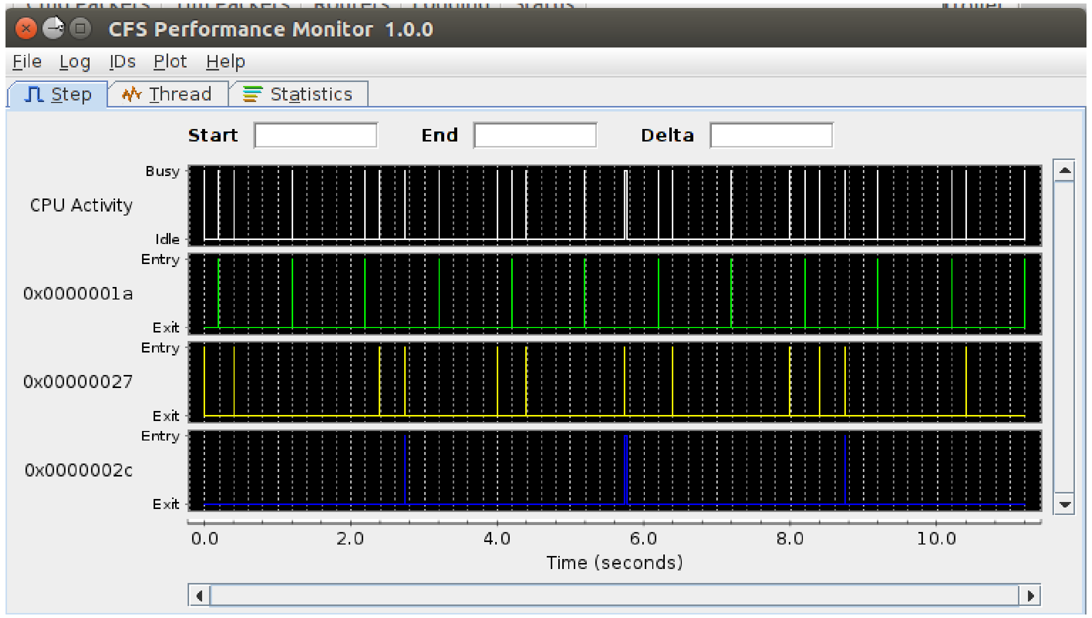
PUT_FILE_COUNT: 0 GET_FILE_COUNT: 0

Ground Working Directory

Flight Working Directory

Flight Event Messages

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



CFS Performance Monitor 1.0.0

Extending OSK

Benchmark App

Coming Soon…

PiSat Control

This requires a PiSat which is currently not in the public domain

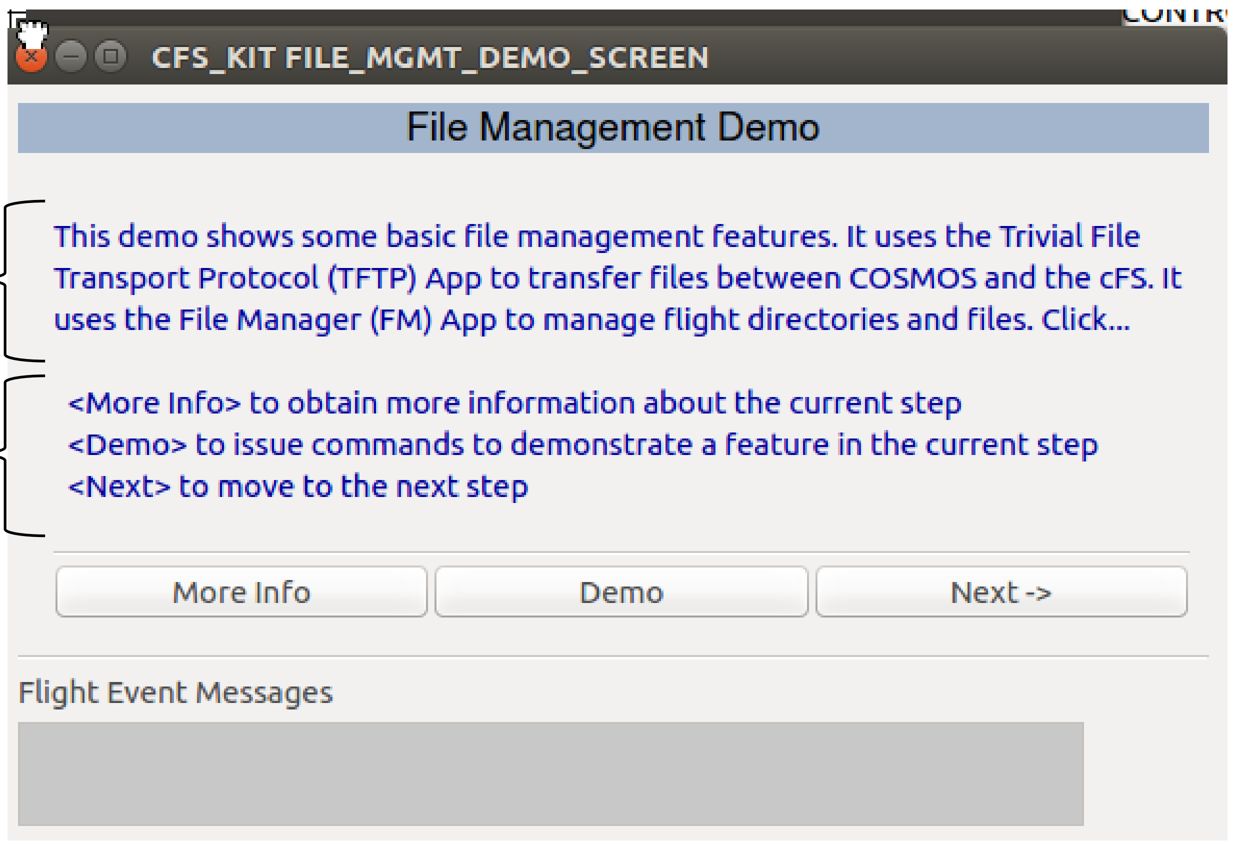
Demos

Screen-based Demos: FM Example (1 of 2)

Each demo follows a common user screen configuration

Description of current step

Button usage description

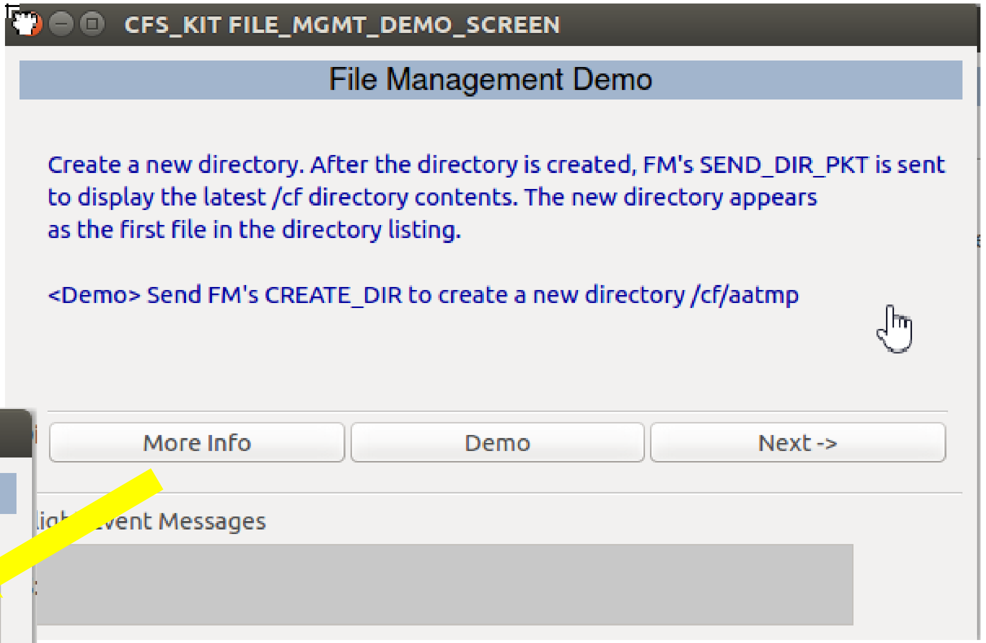


This demo shows some basic file management features. It uses the Trivial File Transport Protocol (TFTP) App to transfer files between COSMOS and the cFS. It uses the File Manager (FM) App to manage flight directories and files. Click...

<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

Screen-based Demos: FM Example (2 of 2)

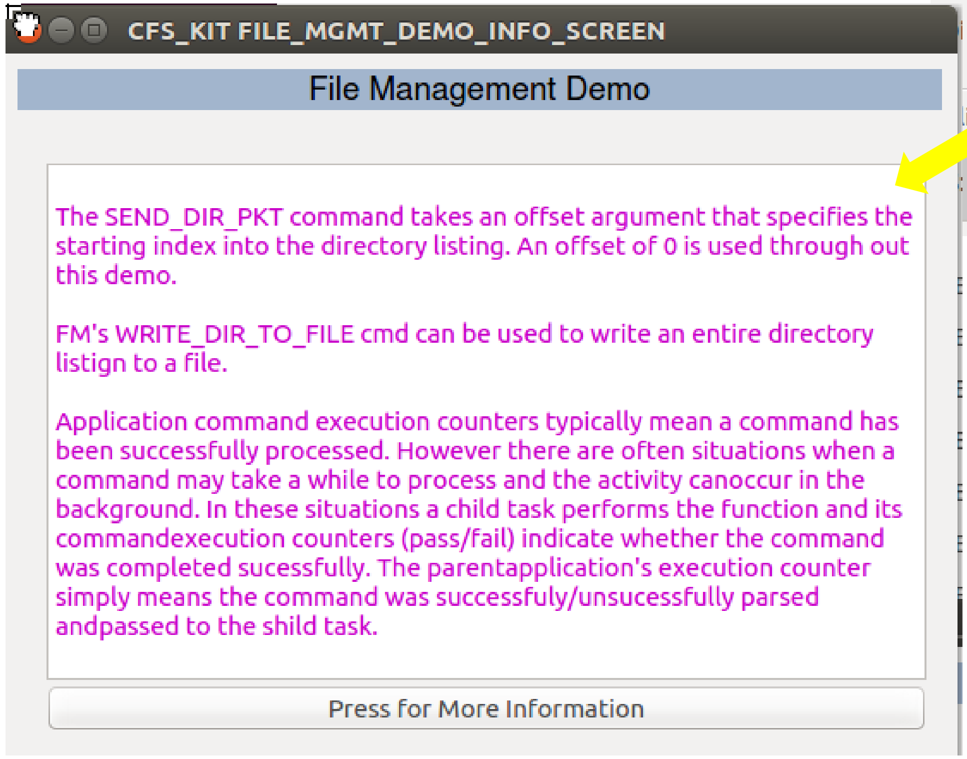
<More Info> provides detailed context-specific information


CFS_KIT FILE_MGMT_DEMO_SCREEN

File Management Demo

Create a new directory. After the directory is created, FM's SEND_DIR_PKT is sent to display the latest /cf directory contents. The new directory appears as the first file in the directory listing.

<Demo> Send FM's CREATE_DIR to create a new directory /cf/aatmp


CFS_KIT FILE_MGMT_DEMO_INFO_SCREEN

File Management Demo

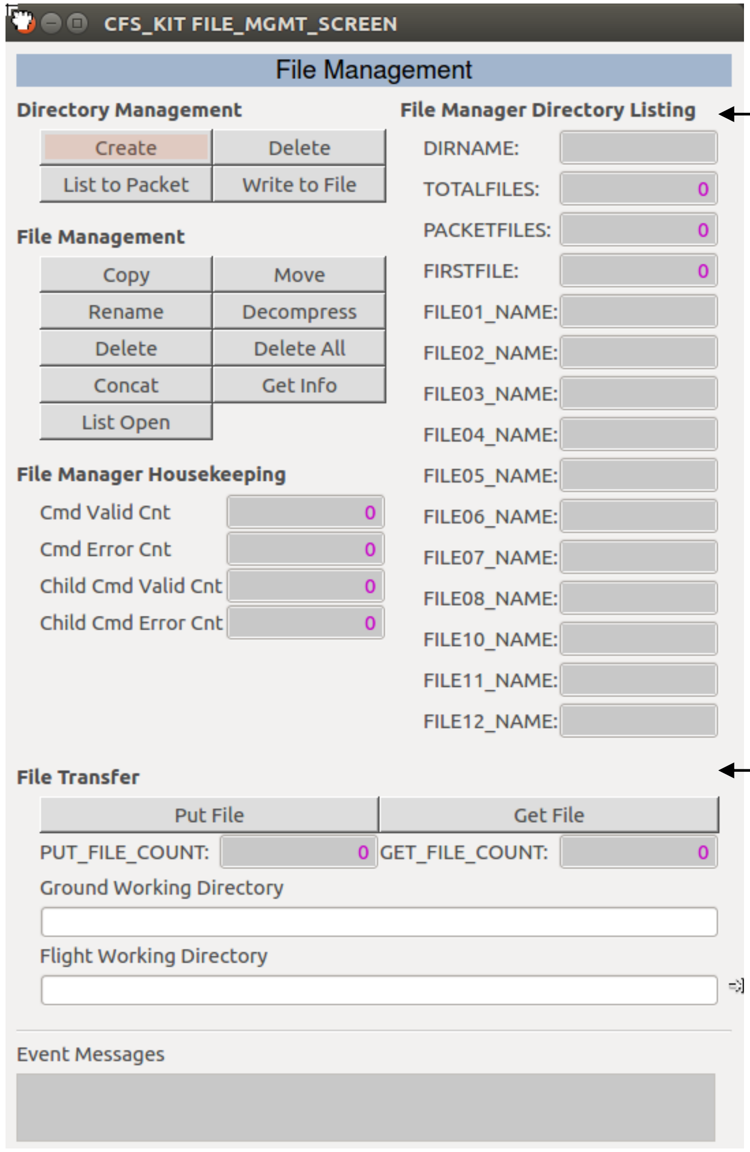
The SEND_DIR_PKT command takes an offset argument that specifies the starting index into the directory listing. An offset of 0 is used through out this demo.

FM's WRITE_DIR_TO_FILE cmd can be used to write an entire directory listign to a file.

Application command execution counters typically mean a command has been successfully processed. However there are often situations when a command may take a while to process and the activity canoccur in the background. In these situations a child task performs the function and its commandexecution counters (pass/fail) indicate whether the command was completed sucessfully. The parentapplication's execution counter simply means the command was successfuly/unsucessfully parsed andpassed to the shild task.

Application Functional Screens

File Management


CFS_KIT FILE_MGMT_SCREEN

File Management

Directory Management

Create | Delete | List to Packet | Write to File

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

File Manager Directory Listing

DIRNAME:	
TOTALFILES:	0
PACKETFILES:	0
FIRSTFILE:	0
FILE01_NAME:	
FILE02_NAME:	
FILE03_NAME:	
FILE04_NAME:	
FILE05_NAME:	
FILE06_NAME:	
FILE07_NAME:	
FILE08_NAME:	
FILE10_NAME:	
FILE11_NAME:	
FILE12_NAME:	

File Transfer

Put File | Get File

PUT_FILE_COUNT: 0 GET_FILE_COUNT: 0

Ground Working Directory

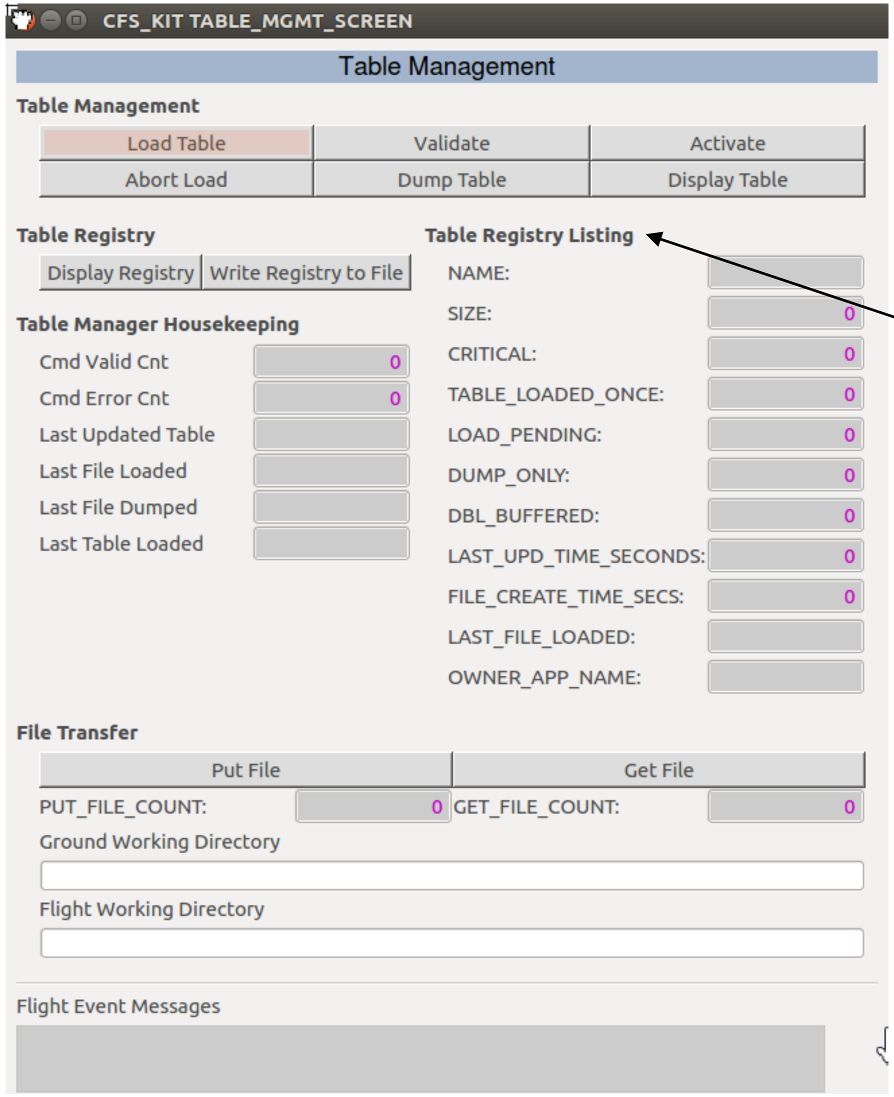
Flight Working Directory

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

Table Management



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: 0 GET_FILE_COUNT: 0

Ground Working Directory

Flight Working Directory

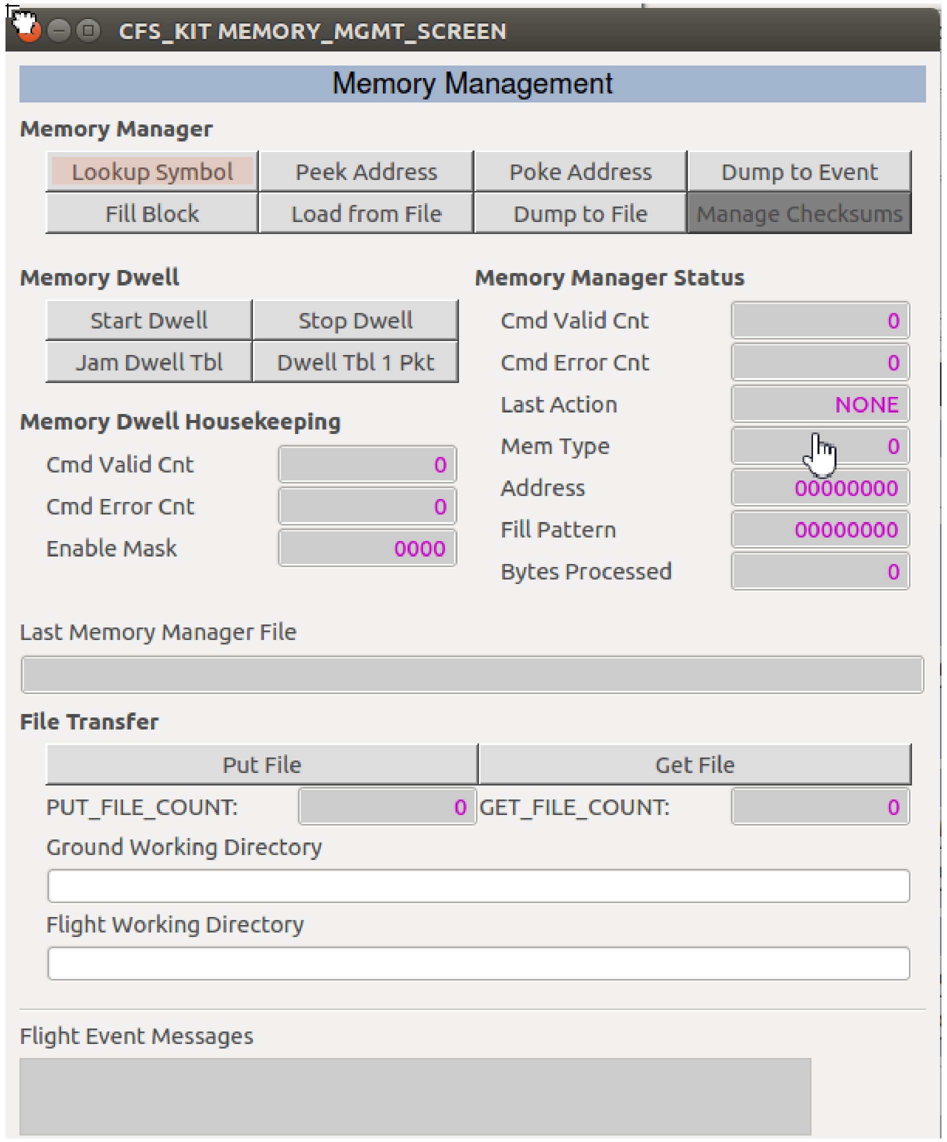
Flight Event Messages

- 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.

Memory Management



CFS_KIT MEMORY_MGMT_SCREEN

Memory Management

Memory Manager

Memory Dwell

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_COUNT: 0 GET_FILE_COUNT: 0

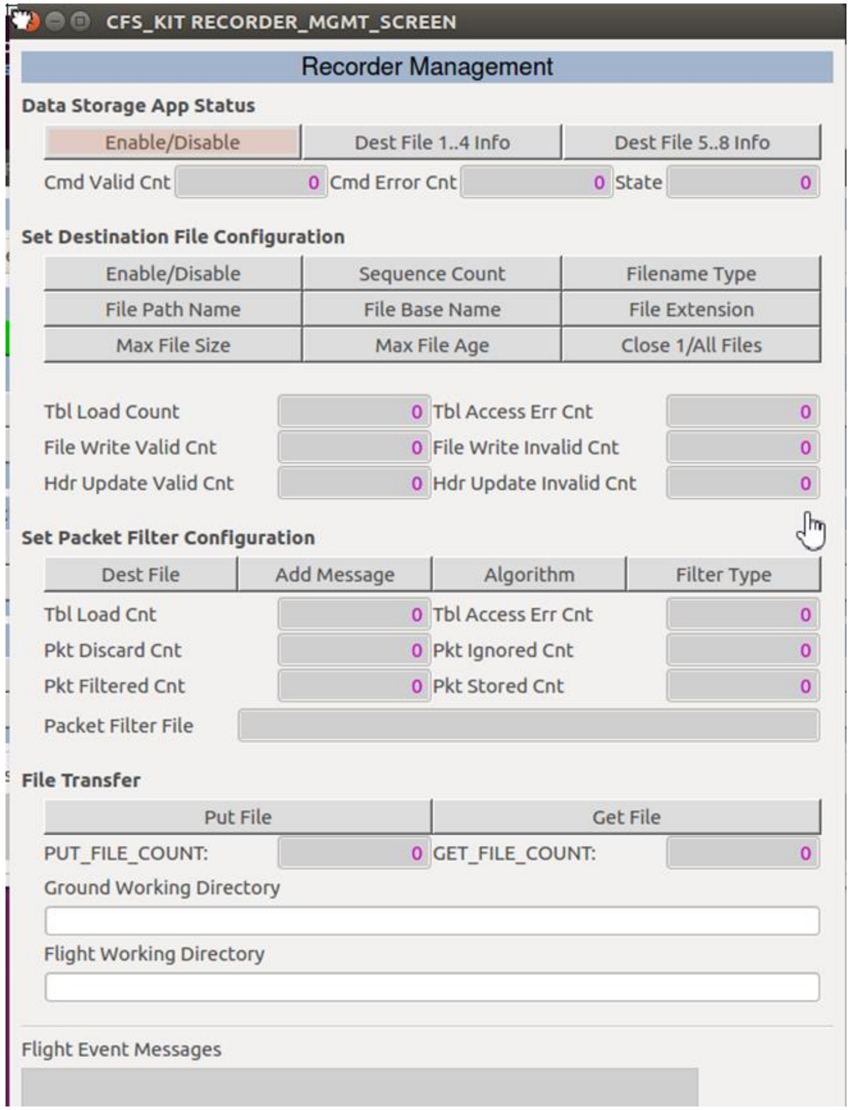
Ground Working Directory

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

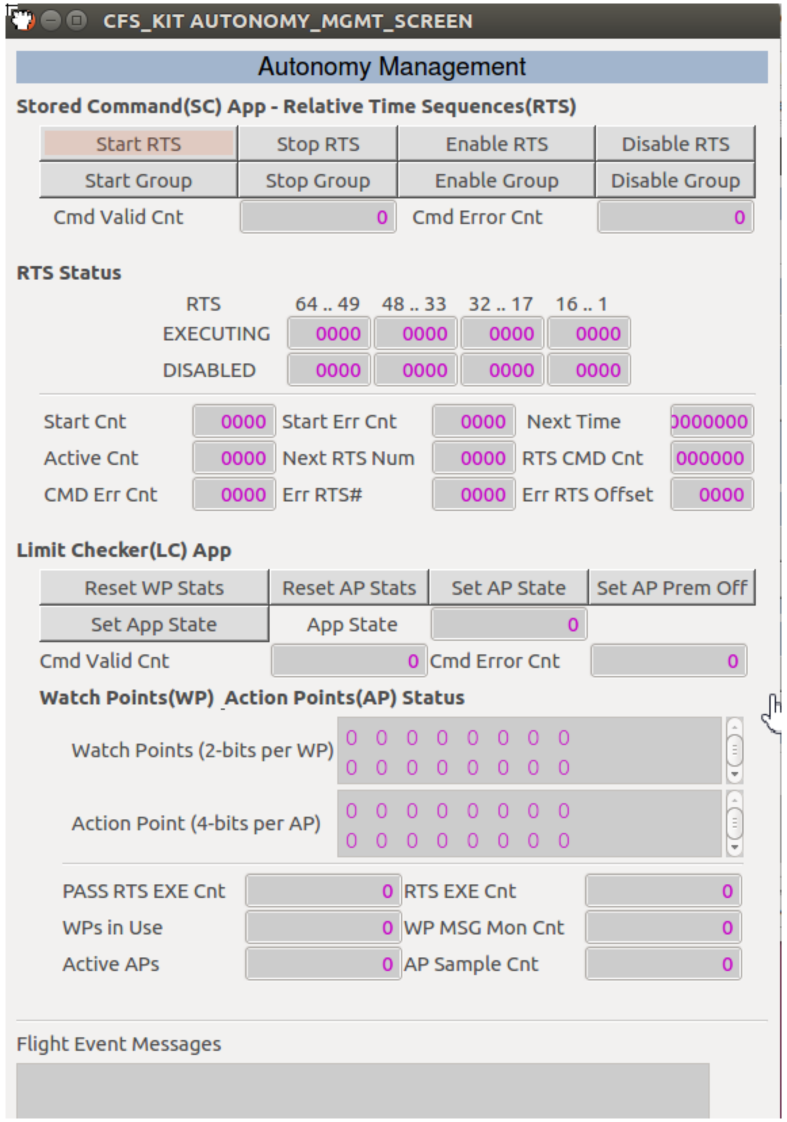
PUT_FILE_COUNT:	0	GET_FILE_COUNT:	0

Ground Working Directory

Flight Working Directory

Flight Event Messages

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

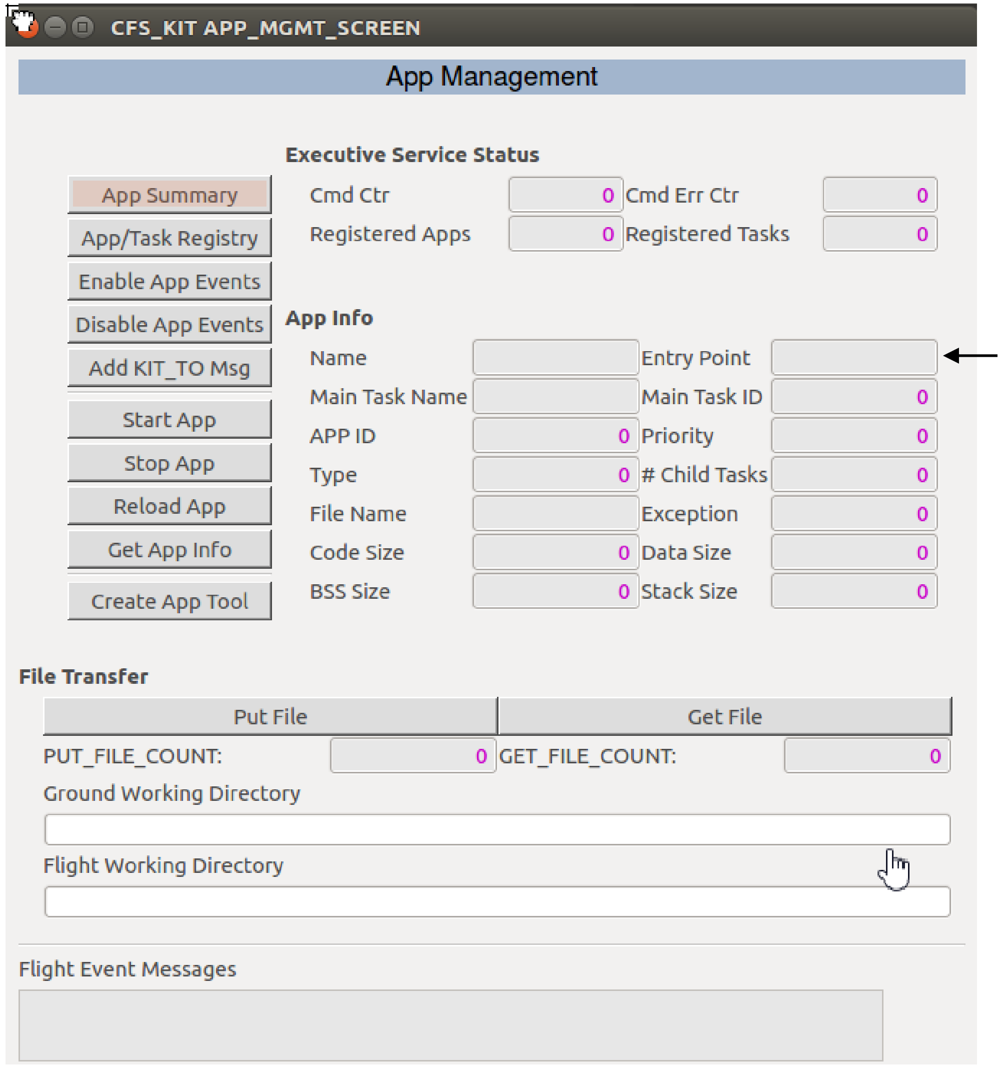
Watch Points (2-bits per WP)
```
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
```

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

Application Management



CFS_KIT APP_MGMT_SCREEN

App Management

App Summary
App/Task Registry
Enable App Events
Disable App Events
Add KIT_TO Msg
Start App
Stop App
Reload App
Get App Info
Create App Tool

Executive Service Status

Cmd Ctr	0	Cmd Err Ctr	0
Registered Apps	0	Registered Tasks	0

App Info

Name		Entry Point	
Main Task Name		Main Task ID	0
APP ID	0	Priority	0
Type	0	# Child Tasks	0
File Name		Exception	0
Code Size	0	Data Size	0
BSS Size	0	Stack Size	0

File Transfer

Put File | Get File

PUT_FILE_COUNT:	0	GET_FILE_COUNT:	0

Ground Working Directory

Flight Working Directory

Flight Event Messages

- *<Get App Info>* commands cFE executive services to send a telemetry packet with the command-specified app

- *<App/Task Registry>* commands cFE executive services to write app or task information to a file that can be transferred to ground via a *<Get File>*

COSMOS Appendix

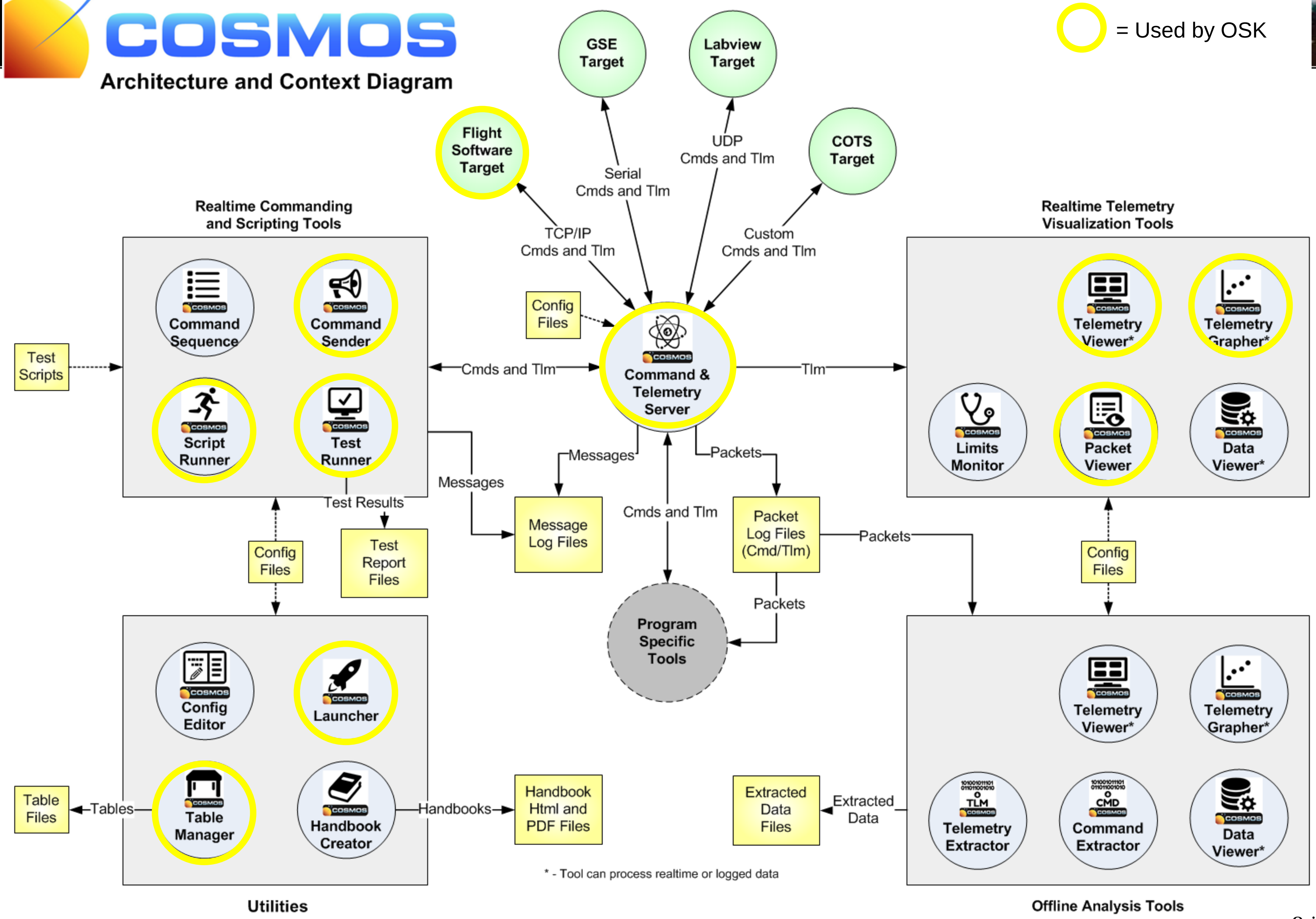
COSMOS Tool Summary (1 of 2)

- **Launcher**
  - Provides a graphical interface for launching each of the tools that make up the COSMOS system
  - *Custom OSK ICON "cFS Starter Kit" launches OSK's main page*
- **Command and Telemetry Server**
  - Connects COSMOS to targets for real-time commanding and telemetry processing.
  - All real-time COSMOS tools communicate with targets through the Command and Telemetry Server ensuring that all communications are logged.
  - Localhost 127.0.0.1 used as cFS connection Targets created
- **Telemetry Viewer**
  - Provides a way to organize telemetry points into custom "screens" that allow for the creation of unique and organized views of telemetry data.
- **Command Sender**
  - Individually send any FSW command using GUI form
  - Raw data files can be used to inject faults
  - *OSK provides custom menus for common cFS commands*
- **Packet Viewer**
  - View any telemetry packet with no extra configuration necessary
  - *OSK provides custom telemetry screens functionally organized*

COSMOS Tool Summary (2 of 2)

- **Telemetry Grapher**
  - Real-time or offline graphing of any FSW telemetry point
  - *OSK provides convenient access through some of its custom screens*
- **Table Manager**
  - Edit and display binary files
  - *OSK provides definitions for most of the cFE binary files and a limited number of cFS application binary files*
- **Script Runner**
  - Develop and execute test procedures using Ruby Scripts and COSMOS APIs
  - *OSK provides additional APIs for functions like file transfer and binary file management*
- **Test Runner**
  - Test framework for organizing, executing, and verifying test scripts
  - *Currently OSK only includes some prototype scripts. The goal is to provide a complete test suite that can be extended by the user.*

COSMOS Architecture and Context Diagram


○ = Used by OSK
GSE Target
Labview Target
Flight Software Target
COTS Target
Serial Cmds and Tlm
UDP Cmds and Tlm
TCP/IP Cmds and Tlm
Custom Cmds and Tlm
Realtime Commanding and Scripting Tools
Command Sequence
Command Sender
Script Runner
Test Runner
Test Scripts
Config Files
Cmds and Tlm
Command & Telemetry Server
Tlm
Realtime Telemetry Visualization Tools
Telemetry Viewer*
Telemetry Grapher*
Limits Monitor
Packet Viewer
Data Viewer*
Messages
Packets
Test Results
Message Log Files
Packet Log Files (Cmd/Tlm)
Test Report Files
Config Files
Program Specific Tools
Config Editor
Launcher
Table Files
Tables
Table Manager
Handbook Creator
Handbooks
Handbook Html and PDF Files
Extracted Data Files
Extracted Data
Telemetry Extractor
Command Extractor
* - Tool can process realtime or logged data
Utilities
Offline Analysis Tools