Event Services

August 2025

Audience & Prerequisites

- **Objectives**
  - Provide a comprehensive description of the cFS Framework Executive Service
- **Intended audience**
  - Software engineers developing with the cFS
- **Prerequisites**
  - cFS Framework Introduction
- **Refer to the Executive Services module for a list of topics covered in each cFE service training module**

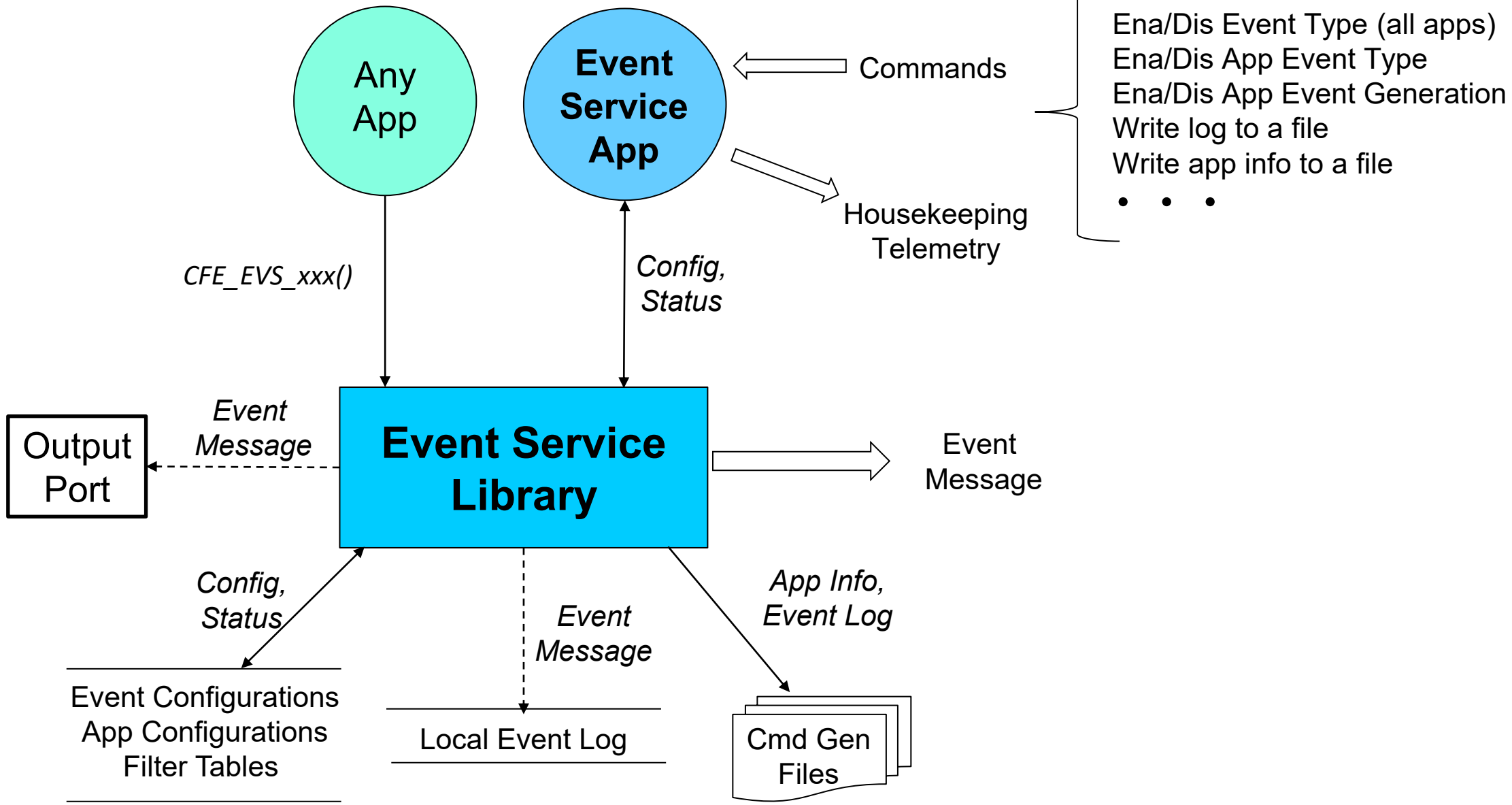
Blue screens contain hands on cFS Basecamp exercises

- Basecamp is a lightweight environment with built-in tutorials for learning the cFS
- https://github.com/cfs-tools/cfs-basecamp

Event Services Overview

- **Provides an interface for sending time-stamped text messages on the software bus**
  - Considered asynchronous because they are not part of telemetry periodically generated by an application
  - Identifiers managed within each application
  - Optionally logged to a local event log
  - Optionally output to a hardware port
- **Four event types defined**
  - Debug, Informational, Error, Critical
- **Event message control**
  - Apps can filter individual messages based on identifier
  - Enable/disable event types at the processor and application scope

Event Services Context



Event Message Format (1 of 3)

Example event message as displayed by a ground system

- Ground systems may choose which event message fields to display
- The Spacecraft ID (defined in cfe_mission_cfg.h) is available but not shown in this example

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Spacecraft time

- System time when the event occurred

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Event type

- Debug, Informational, Error, Critical

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Event Message Format (2 of 3)

Processor ID

- Defined in cfe_platform_cfg.h

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Application name

- cFE Service name or app name as defined in *cfe_es_startup.scr*

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Event ID

- Managed within an application

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Event text

- Created using printf() format options

14:14:40.500 ERROR CPU=CPU3 APPNAME=CFE_TBL EVENT ID=57 Unable to locate 'TST_TBL.invalid_tbl_02' in Table Registry

Event Message Format (3 of 3)

- **This is an example of a *"Long Format"* event message which is the cFE default**

- **A *"Short Format"* platform option is available that allows messages to be sent without the text portion**
  - This is useful in low telemetry bandwidth situations
  - An event message ID lookup tool would simplify operations so ground operators could correlate event IDs with the event text

Event Message Filtering (1 of 2)

- **"Filter Mask"**
  - Used to determine when to send a filtered event
  - A bit-wise Boolean AND is performed on the event ID message counter and if the result is zero then the event is sent
  - Mask applied before the sent counter is incremented

- **"Filter Mask" Examples**
  - 0x0000 => Every message sent
  - 0x0003 => Every 4th message sent
  - 0xFFFE => Only first two messages sent

- **Reset filter**
  - Filters can be reset from an application or via command

⚠ Requires knowledge of binary and hexadecimal numbering systems

Event Message Filtering (2 of 2)

- **The Software Bus 'No Subscriber' event message, Event ID 14**
  - See *cfe_platform_cfg.h* CFE_SB_FILTERED_EVENT1
  - Default configuration is to only send the first 4 events
    - Filter Mask = 0xFFFC
- **CFE_EVS_MAX_FILTER_COUNT (cfe_evs_task.h) defines maximum count for a filtered event ID**
  - Once reached event becomes locked
  - Prevents erratic filtering behavior with counter rollover
  - Ground can unlock filter by resetting or deleting the filter

Event Message Control

- **Processor scope**
  - Command allows events to be enabled/disabled based on type
    - Debug, Information, Error, Critical
- **Application scope**
  - Commands allow events to be enabled/disabled based on type
  - Commands allow all events to be enabled/disabled
- **Event message scope**
  - During initialization apps can register events for filtering for up to CFE_EVS_MAX_EVENT_FILTERS defined in *cfe_platform_cfg.h*
  - Command allows event filter masks to be modified

Reset Behavior

- **Power-on Reset**
  - No data preserved
  - Application initialization routines register with the service
  - If configured local event log enabled

- **Processor Reset**
  - If the cFS target is configured with an event log, the following is preserved:
    - Messages
    - Mode: Discard or Overwrite
    - Log Full and Overflow status

Retrieving Onboard State

- **Housekeeping Telemetry**
  - Log Enabled, Overflow, Full, Enabled
  - For each App: AppID, Events Sent Count, Enabled

- **Write application data to file. For each app**
  - Active flag – Are events enabled
  - Event Count
  - For each filtered event
    - Event ID
    - Filter Mask
    - Event Count – Number of times Event ID has been issued

- **Local event log**
  - If enabled events are written to a local buffer
  - Log “mode” can be set to over write or discard
  - Serves as backup to onboard-recorder during initialization or error scenarios
  - Suitable for multi-processor architectures
  - Command to write log to file

System Design: Guidelines

- **DEBUG logging level should be disabled in flight**
- **Telemetry Output should subscribe to and downlink event messages**
- **Local event log**
  - Suitable for multi-processor architectures
  - Serves as backup to onboard-recorder during initialization or error scenarios
  - Preserved across processor resets
- **Event message guidelines**
  - Don’t desensitize operators with too many events
    - Be judicious and consistent with how and when informational events are used
    - Consider whether routine telemetry is a better option for certain onboard state knowledge
  - Balance testing and operational needs
    - Is the event a convenience for testing or does it help operations?
  - Be cognizant that other apps can monitor events and take corrective action based on events

Application Development: Guidelines

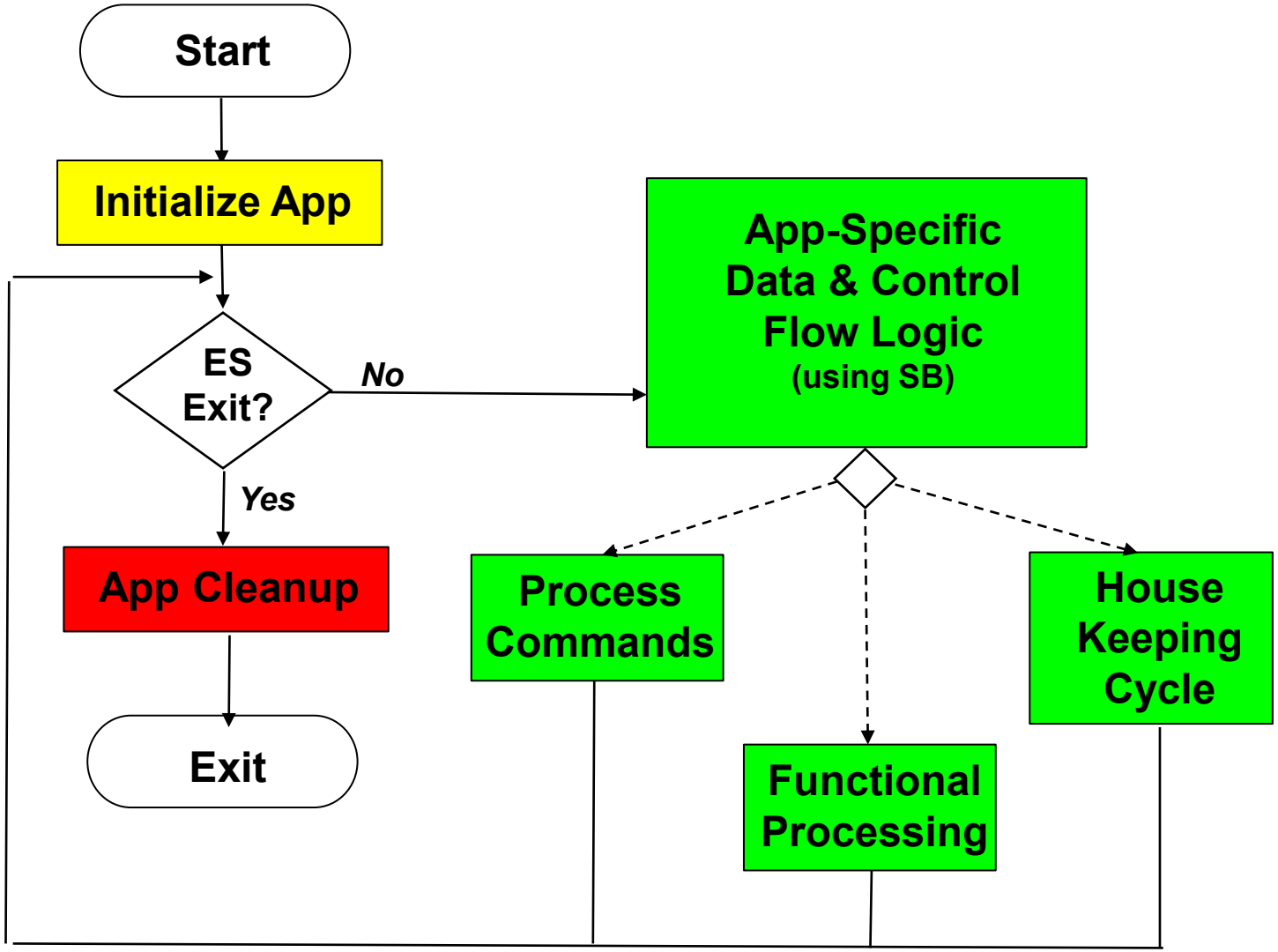
- **Applications should register with Event Services immediately after registering the app with Executive Services so events can be used rather than syslog writes**
  - Apps should write to the ES system log if event services cannot be registered
  - An app must register with event services before it can send any events
  - Calls to CFE_EVS_SendEvent() will have no effect if the app is not registered with EVS

- **cFE libraries cannot register with EVS**
  - Library events are sent in the context of the app calling the library function

- **Event Filtering**
  - Apps should limit the amount of filtering done within the app (ground should have ultimate control over filtering)
  - Apps should avoid “spamming” event messages

- **App Development**
  - Any app can subscribe to event messages (like any other software bus message)

App Development: Example Control Flow



App Development: Example Control Flow EVS Calls

Initialize App

CFE_EVS_Register() – Register app with Event Services includes an optional event filter table pointer parameter

All nominal processing flow

CFE_EVS_SendEvent() – Send event messages when needed

Housekeeping Cycle

CFE_EVS_ResetFilter() – Rarely used and typically in interface apps where the event situation may be corrected

App Cleanup

CFE_ES_ExitApp() – Unregisters app from Event Services

APIs

Registration APIs	Purpose
CFE_EVS_Register	Register an application for receiving event services

Send Event APIs	Purpose
CFE_EVS_SendEvent	Generate a software event.
CFE_EVS_SendEventWithAppID	Generate a software event given the specified Application ID.
CFE_EVS_SendTimedEvent	Generate a software event with a specific time tag.

Reset Event Filter APIs	Purpose
CFE_EVS_ResetFilter	Resets the calling application's event filter for a single event ID.
CFE_EVS_ResetAllFilters	Resets all of the calling application's event filters.

Command List (1 of 2)

Command List	Purpose
CFE_EVS_NoopCmd	This function processes "no-op" commands received on the EVS command pipe
CFE_EVS_ClearLogCmd	This function processes "clear log" commands received on the EVS command pipe
CFE_EVS_ReportHousekeepingCmd	Request for housekeeping status telemetry packet
CFE_EVS_ResetCountersCmd	This function resets all the global counter variables that are part of the task telemetry
CFE_EVS_SetFilterCmd	This routine sets the filter mask for the given event_id in the calling task's filter array
CFE_EVS_EnablePortsCmd	This routine sets the command given ports to an enabled state
CFE_EVS_DisablePortsCmd	This routine sets the command given ports to a disabled state
CFE_EVS_EnableEventTypeCmd	This routine sets the given event types to an enabled state across all registered applications
CFE_EVS_DisableEventTypeCmd	This routine sets the given event types to a disabled state across all registered applications
CFE_EVS_SetEventFormatModeCmd	This routine sets the Event Format Mode
CFE_EVS_EnableAppEventTypeCmd	This routine sets the given event type for the given application identifier to an enabled state

Command List (2 of 2)

Command List	Purpose
CFE_EVS_DisableAppEventTypeCmd	This routine sets the given event type for the given application identifier to a disabled state
CFE_EVS_EnableAppEventsCmd	This routine enables application events for the given application identifier
CFE_EVS_DisableAppEventsCmd	This routine disables application events for the given application identifier
CFE_EVS_ResetAppCounterCmd	This routine sets the application event counter to zero for the given application identifier
CFE_EVS_ResetFilterCmd	This routine sets the application event filter counter to zero for the given application identifier and event identifier
CFE_EVS_ResetAllFiltersCmd	This routine sets all application event filter counters to zero for the given application identifier
CFE_EVS_AddEventFilterCmd	This routine adds the given event filter for the given application identifier and event identifier
CFE_EVS_DeleteEventFilterCmd	This routine deletes the event filter for the given application identifier and event identifier
CFE_EVS_WriteAppDataFileCmd	This routine writes all application data to a file for all applications that have registered with the EVS
CFE_EVS_SetLogModeCmd	Sets the logging mode to the command specified value.
CFE_EVS_WriteLogDataFileCmd	Requests the Event Service to generate a file containing the contents of the local event log.

Platform Configuration Parameters

Parameter	Purpose
CFE_PLATFORM_EVS_START_TASK_PRIORITY	Define EVS Task Priority
CFE_PLATFORM_EVS_START_TASK_STACK_SIZE	Define EVS Task Stack Size
CFE_PLATFORM_EVS_MAX_EVENT_FILTERS	Define Maximum Number of Event Filters per Application
CFE_PLATFORM_EVS_DEFAULT_LOG_FILE	Default Event Log Filename
CFE_PLATFORM_EVS_LOG_MAX	Maximum Number of Events in EVS Local Event Log
CFE_PLATFORM_EVS_DEFAULT_APP_DATA_FILE	Default EVS Application Data Filename
CFE_PLATFORM_EVS_PORT_DEFAULT	Default EVS Output Port State
CFE_PLATFORM_EVS_DEFAULT_TYPE_FLAG	Default EVS Event Type Filter Mask
CFE_PLATFORM_EVS_DEFAULT_LOG_MODE	Default EVS Local Event Log Mode
CFE_PLATFORM_EVS_DEFAULT_MSG_FORMAT_MODE	Default EVS Message Format Mode

Mission Configuration Parameters

Parameter	Purpose
CFE_MISSION_EVS_MAX_MESSAGE_LENGTH	Maximum Event Message Length

Event Services Exercises - 1

1. TODO

2. TODO

Power on Reset