NASA GSFC FLIGHT SOFTWARE SYSTEMS BRANCH

FSW VERSION DESCRIPTION DOCUMENT

CFS FILE MANAGER (FM) APPLICATION

BUILD: FM 2.6.0

RELEASE DATE: 09/27/2021

1.0 FSW VERSION DESCRIPTION

1.1 PURPOSE AND SUMMARY

The purpose of this build is to continue to refine the cFS File Manager (FM) application product. This build provides various bug fixes and enhancements but does not include any new functionality. The primary purpose of this release is to ensure compatibility between the FM application and cFS Caelum.

This document serves as the notification of the Build 2.6.0 release of the cFS FM application.

1.2 NEW/CHANGED FUNCTIONALITY IN THIS VERSION

Table 1.2-1 identifies the DCRs that have been implemented in this FSW version. For each DCR the "Key" column shows the corresponding DCR in the GSFC cFS tracking system.

Table 1.2-1 – DCRs Implemented in this Version

Key	Summary	Description
GSFCCFS-1088	Migrate FM unit tests to distributed UT Assert	
GSFCCFS-1146	FM may be impacted by OSAL ticket 344	https://github.com/nasa/osal/issues/344
GSFCCFS-1180	FM has static code analyis findings	In analysis done on 7/10/2020, CodeSonar flagged the attached findings.
GSFCCFS-1250	FM FileInfo Command Does Not Acquire File Permissions	The FM command "FM_GetFileInfoCmd" is required by 2011 to report for each file "the file mode (permissions), as a 4-byte value" in addition to the file size and last time modified. It does not appear to report the file mode when requested.
GSFCCFS-1251	FM_ChildSetPermissionsCmd does not update ChildCC fields in FM_GlobalData	All of the functions in the FM childtask (fm_child.c) utilize two fields from the FM_GlobalData structure to denote the current and previous commands executed. The current command is denoted by FM_GlobalData.ChildCurrentCC. The previous command is denoted by FM_GlobalData.ChildPreviousCC. At the end of each childtask command function, FM_GlobalData.ChildCurrentCC is set to 0 indicating completion. FM_GlobalData.ChildPreviousCC is set to the command code for the command that was executed. This is not present in the function FM_ChildSetPermissionsCmd, which handles setting file permissions.

GSFCCFS-1257	Update FM to use new cFE Message Module	
GSFCCFS-1402	FM4000 does not match the code	Requirement FM4000 states that the FM housekeeping packet shall include: "c) For each file system: Total number of open files " In the code, this information is not present in the housekeeping packet, and it doesn't look like this exact information is currently reported anywhere (free space and open files are reported, but number of open files per file system is not).
GSFCCFS-1477	FM does not build with eval-cert3	
GSFCCFS-1579	FM doxygen config file should be renamed for clarity	The filename "fm_config.txt" suggests that this a configuration file for the app itself as opposed to a configuration file for doxygen.
GSFCCFS-1588	FM should use const for function arguments where possible	
GSFCCFS-1621	FM event messages do not allow for extended message IDs	Events that print out a messageID value use the 0x04X format specifier, which does not work for longer message IDs.
GSFCCFS-1708	FM should not pend forever on the software bus	
GSFCCFS-1724	FM should use strict resource IDs	

1.3 MISSING PLANNED FEATURES AND KNOWN PROBLEMS

Table 1.3-1 identifies currently open DCRs that are not addressed in this build. Any workarounds that may apply are identified.

Information on currently open DCRs is available at:

https://etdjira.gsfc.nasa.gov/projects/GSFCCFS/issues

Note that this is a restricted website that requires a server account. Additional DCRs may have been submitted after preparation of this VDD. A cFS FM DCR report containing a listing of open DCRs is available upon request for customers who do not have access to the restricted server. Please contact the cFS Program Team, cfs-program@nasa.onmicrosoft.com.

Table 1.3-1 – Currently open DCRs

Key	Summary	Description
GSFCCFS-1032	FM return statements not needed for void function	Finding from JSC code review
GSFCCFS-1407	Add Signature Checking Command to FM	Add a command to check the signature of a file. In the open source version of the app, this will call an empty stub in fs_lib that will always succeed (allows external users to fill in their own implementation). In the gateway version, an actual implementation of the signature verification will be provided.
GSFCCFS-1121	FM should use OSAL functions to verify filenames	Describe the bug A few commands to the cFE core services, and possibly some cFS applications, are able to create files that cannot then be moved, renamed, deleted, or otherwise by FM. One example is EVS_WRITELOG2FILE (CFE_EVS_FILE_WRITE_LOG_DATA_CC). This command (and all others that take filenames in the cFE core) don't check the validity of a filename; they simply pull the filename from the command and call OS_creat() or OS_open() immediately. FM, on the other hand, calls CFS_IsValidFilename() to check for validity of provided filenames in all commands. So, if a command like EVS_WRITELOG2FILE is used to create a file with an invalid filename, FM cannot then access that file in any way.
GSFCCFS-1062	FM configuration parameter limits need clarification	A number of FM configuration parameters have limits for which the reason is obscure at best. Limits need to be re-evaluated and comments should give clear reasoning for the limit.
GSFCCFS-1058	FM_DELETE_INT_CC appears to be redundant with FM_DELETE_CC	The doxygen comment for FM_DELETE_INT_CC states: "This is a special version of the #FM_DELETE_CC command for use when the command is sent by another application, rather than from the ground. This version of the command will not generate a success event, nor will the command increment the command success counter. The intent is to avoid confusion resulting from telemetry representing the results of delete commands sent by other applications and those sent from the ground."

		However, this does not appear to be the case. Both FM_DELETE_INT_CC and FM_DELETE_CC call the same functions. It appears that FM_DELETE_INT_CC does increment the command success counter, but does not send an event message from the child task. I think that the need for FM_DELETE_INT_CC needs to be reevaluated and if it is needed, it should be updated to match its description.
GSFCCFS-965	Replace FM Internal Command Code with Internal MID instead	Currently, the FM app has an "internal" command code defined for a delete file request that originates from another app instead of the ground. However, the definition for that command code (FM_DELETE_INT_CC) is located in a the header file fm_msgdefs.h, which is located in the app's "fsw/src" directory, and strictly speaking, not accessible by other apps. Instead of using an internal command code, FM could use an internal message ID (e.g., FM_INTERNAL_CMD_MID) that can be defined in fm_msgids.h, which is located in the platform_inc directory. Other cFS apps could then access that command MID and send the internal delete command to the FM app without reaching into what should be an FM-local header file.
GSFCCFS-1083	Add untar command to FM	WFIRST has requested that untar capability be added in cFE. In design discussions with the framework team, it was decided that it made sense to pull the decompress capability out of the framework and into a library, and to then add an untar command to the FM app.
GSFCCFS-1026	Reduce redundant code in fm_cmd_utils.c	Most verify functions in fm_cmd_utils contain several instances of checking the FilenameState. Could that code be refactored into its own function that receives the set of valid return codes, and is able to validate the return code or report the errors (maybe a bitmap). This would eliminate a big portion of redundant code. Or could it use switch statements?

2.0 DELIVERED PRODUCTS

Table 2-1 identifies the locations of FSW products relevant to this FSW Build. The version or date of the Build and where the product can be located are provided. Changes from a previous VDD are identified.

Table 2-1 – Delivered Products and their Locations

Software Element	Changed with this Version?	New Version or Date	Location
Source Code of this FSW Build	Yes	2.6.0	https://github.com/nasa/fm
Doxygen Documentation	Yes	N/A	https://github.com/nasa/fm
Unit Test Data	Yes	2.6.0	https://github.com/nasa/fm
FSW Make Files	Yes	2.6.0	https://github.com/nasa/fm

3.0 INSTALLATION PROCEDURES

In order to build and install the FM application, it must be added to the cFE CMake build system. This is done by modifying the TGTX_APPLIST in the cFE targets.cmake file. This is shown in the trivial example below.

```
SET(TGT1_NAME cpu1)
SET(TGT1_APPLIST cfs_lib fs_lib fm)
SET(TGT1_FILELIST cfe_es_startup.scr)
```

After FM is added to the targets.cmake file, it is built and installed using the standard cFE CMake build instructions. These instructions are available in cFE CMake documentation:

https://github.com/nasa/cFE/blob/main/cmake/README.md

4.0 CONFIGURATION SUMMARY AND VERSION IDENTIFICATION

This software can be found in the FM GitHub repository (https://github.com/nasa/FM) under the tag "2.6.0".

Verification of the version can be done by sending an FM NOOP command which produces an event message containing the version information. In addition, the initialization event message generated during the application startup provides the version information.

ACRONYMS

ACS	Attitude Control System
C&DH	Command and Data Handling
cFS	Core Flight System
CM	Configuration Management
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
DCR	Discrepancy/Change Request
ETU	Engineering Test Unit
FM	File Manager
FSB	Flight Software Branch
FSW	Flight Software
GSFC	Goddard Space Flight Center
I&T	Integration & Test
JSC	Johnson Space Center
POSIX	Portable Operating System Interface
RTOS	Real-Time Operating System
SMP	Symmetric Multiprocessing
T&C	Telemetry and Command
TBD	To Be Determined
URL	Universal Resource Locator
VDD	Version Description Document