NASA/CR-20205010026

Open Source Core Flight System (cFS) Flight Software (FSW) Verification & Validation (V&V) Final Summary

IV&V Analysis Technical Report

John W Bradbury

Technical Monitor: Scott Benton

December 2020

NASA STI Program ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at http://www.sti.nasa.gov

- E-mail your question to help@sti.nasa.gov

- Phone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Information Desk
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA/CR-20205010026

Open Source Core Flight System (cFS) Flight Software (FSW) Verification & Validation (V&V) Final Summary

IV&V Analysis Technical Report

John W Bradbury, Engility (TASC)
NASA Independent Verification and Validation Facility, Fairmont, WV

Technical Monitor: Scott Benton
NASA Independent Verification and Validation Facility, Fairmont, WV

National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

December 2020

Level of Review: This material has been technically reviewed by technical management.

Available from

NASA STI Program
Mail Stop 148
NASA's Langley Research
Center Hampton, VA
23681-2199

National Technical Information
Service 5285 Port Royal Road
Springfield, VA 22161
703-605-6000

National Aeronautics and Space Administration

Independent Verification and Validation (IV&V)

Open Source Core Flight System (cFS)
Flight Software (FSW) Verification & Validation (V&V)
Final Summary
IV&V Analysis Technical Report

October 30, 2020

Approved By: Scott Benton
Software Assurance Research Program (SARP) Manager
NASA Independent Verification and Validation Facility
100 University Drive, Fairmont WV 26554

TABLE OF CONTENTS

Table of Figures

Table of Tables

This report summarizes the analyses performed as part of the Software Assurance Research Program (SARP) sponsored Open Source Core Flight System (cFS) Flight Software (FSW) Verification & Validation (V&V) project.

The core Flight System is based on Goddard Space Flight Center (GSFC) heritage NASA Class "B"[1] software and has been successfully applied to Class "B" and other less critical use. The cFS architecture and framework approach to FSW is described by its NASA development team as *"a platform and project independent reusable software framework and set of reusable software applications"... "suitable for reuse on any number of NASA flight projects and/or embedded software systems at a significant cost savings".*[2]

The cFS is based on multiple layers of abstractions intended to limit the impact of hardware or software changes while supporting the insertion of mission-specific applications, thus facilitating meaningful formalized software reuse with concomitant improvements in software cost, schedule and quality. A representation of this layered architecture is presented in below.

The identified record for the intended or claimed benefits of the cFS approach has not identified verification and validation reduction as an intended benefit. At the time of the cFS's development and propagation, the high-level NASA expectations such as those found within NASA Procedural Requirements (NPR) -7150.2 "Software Engineering Requirements" included a requirement for off-the-shelf (OTS) software to be verified and validated to the same "level" or "level of confidence" as developed software[3]. Demonstrating the ability to reuse software assurance verification and validation evidence would provide a rigorous and flexible match for the high-level NASA expectation of cost savings and improved scheduled certainty.

[1] "Non-Human Space Rated Software Systems or Large Scale Aeronautics Vehicles" per NPR 7150.2.
[2] https://cfs.gsfc.nasa.gov/
[3] Reference NPR 7150.2 SWE-027 Rev I/R "*Off-the-shelf software is validated to the same level of confidence as would be required of the developed software."* or Rev A, Rev B and Rev C *"The software component is verified and validated to the same level required to accept a similar developed software component for its intended use."*

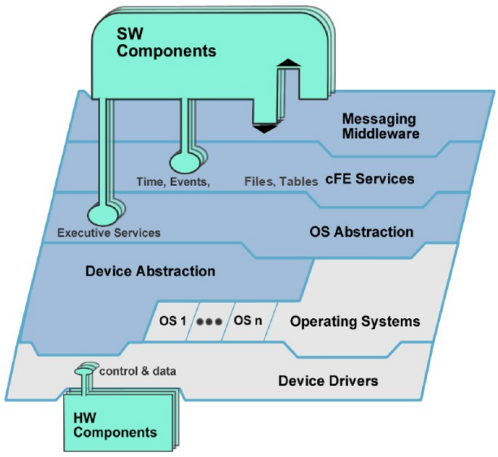


Figure 1 – cFS Layered Service Architecture[4]

The goals for this SARP research project included: (a) the review and analysis of the V&V of NASA's open source core Flight System (cFS) within high-value NASA programs; (b) the identification of reusable program or project cFS V&V elements or assurance evidence; (c) identification of abstraction levels with potential to limit software (SW) assurance effort; and (d) establishment of a template for tailoring cFS V&V using the NASA Independent Verification and Validation (IV&V) verification and validation lifecycle as a baseline.

This SARP project planned to leverage a concurrent IV&V effort on the announced expansion of cFS into Class "A"[5] safety-critical software, particularly the early uncrewed flight of the Orion Camera Controller (CC) and Orion Vision Processing Unit (VPU). The VPU includes the semi-independent Backup Flight Software (BFS) intended to function as the control to common mode software failures within the primary Orion flight software. While executing this SARP project, the Orion Program moved away from the formal verification of the Orion instance of cFS software. cFS was instead accepted for the Class "A" safety-critical Orion Artemis I flight's use based on the overall perception of (a) prior cFS flight success and (b) prior verification and validation[6] completion. SARP Open Source cFS FSW V&V project activities were subsequently reoriented to achieving the following goals: (a) documentation of multiple projects' verification and validation approach, (b) documentation of verification and validation tool reuse, and (c) completion of Klocwork static code analysis (SCA) as a gauge of likely code maturity and dependability including delivery of higher impact SCA issues to the cFS development team. Documented elements of these activities, which are broken into seven tasks described below, include:

- The relative stability/volatility of cFS instances describing the observed changes to cFS software including host operating systems and hardware from build to build and use to use across multiple projects.
    - Task 1: Describe the changes to Core Flight Executive (cFE)/cFS software from build to build

[4] National Aeronautics and Space Administration https://cfs.gsfc.nasa.gov/cFS-OviewBGSlideDeck-ExportControl-Final.pdf
[5] "Human Rated Space Software Systems" per NPR 7150.2.
[6] Preflight verification and validation plans for future missions can be considered to be possible future work.

        - Is there a trend in software changes? --- (which versions used on which missions)
        - This would leverage the Source Forge and github release notes.
        - For the purposes of reusing software assurance evidence, a trend toward small, gradual changes between builds (as opposed to major changes) is desirable.
    - Task 6: Understand perceptions of cFE/cFS stability
        - What is the perception of cFE/cFS stability on NASA projects surveyed?
        - Are projects using the most recent release, or the version the heritage mission reused (or put another way, are NASA projects sufficiently involved in the continuous improvement of cFE/cFS)?
        - How this perception matches with the results of the other tasks will tell us something about the larger question: if people view it as stable, but significant issues continue to be found, and the code changes dramatically over time, this is a problem.
- The observed trends within the V&V approaches from project to project including instances of the reuse of software assurance evidence, tools or processes.
    - Task 2: Assess the evolution of V&V approaches from project to project --- (decide baseline, with respect to SCA, other analysis types)
        - Are projects giving less attention to cFE/cFS verification and validation as time goes on?
        - If the software continues to change, or if significant issues continue to arise, then a growing sense of comfort with cFE/cFS might be a problem.
    - Task 4: Assess the use of heritage assurance evidence used by the projects
        - How did V&V heritage affect the approach?
        - Any objective rationale for doing less than the previous mission would depend, in part, on what was done before.
        - Using heritage assurance evidence is different than believing cFE/cFS is stable.
- The observed trends within identified cFE/cFS-related issues over time including issues identified, resolved and remaining.
    - Task 3: Assess the significance of cFE/cFS-related issues over time
        - If NASA projects are going to do less verification and validation over time on this software, then we would need to see that the 'severity' of issues decrease over time.
        - Also might be interesting to compare these results with Task 1, to see if there is an uptick in issues without an uptick in software changes.
        - In-flight anomaly database as a source for this --- check old IV&V Technical Discussions for anomaly presentation on cFE/cFS.
- Review of the cFS improvement processes within the reusable open source software (OSS) software framework and applications goal.
    - Task 5: Understand the communication of cFE/cFS analysis results/data back to GSFC
        - How were issues, challenges, etc. communicated to GSFC?
        - For continuous improvement to be successful, there must be a feedback loop to GSFC.

  - For continuous improvement to be important, projects have to adopt the new releases.
  - The assumption is that continuous improvement is occurring – this task might indicate otherwise.
- Identify objective indicators of FSW goodness.
  - Task 7: Perform SCA of OSS cFS application and service code.

Summary of Results:

Software costs represent 5% to 15% of program budgets[7], and software testing is a major SW cost driver[8]. Test cost in turn is a major pressure on verification and validation efforts. Repeating testing does not assure improved SW performance. Testing is most effective when tests address changed conditions or changed SW. Surveyed NASA projects displayed reluctance to perform cFS V&V testing, which they presumed would be a repeat of prior testing. The cFS V&V SARP project experience was similar to an earlier "NASA Software Engineering Benchmarking Study" Commercial Off The Shelf (COTS)/Government Off The Shelf (GOTS) observation in this area[9]. It is unlikely that non-value added duplication of testing has been performed due to changes in (a) NASA leadership requirements documented as Software Engineering Requirements (SWEs) within NPR 7150.2, (b) cFS project/program hardware (HW) or Operating System (OS), and cFS project/program software classifications.

The analyses completed were positive and have established a number of observations:

- There are a significant number of projects or programs[10] successfully applying the cFS code as shown in Table 1 below. cFS use spans a very wide range of software classifications and NASA centers. Substantial variations between cFS instances' OS and processing HW have been identified.
  - Adoption of cFS continues; 40 instances have been identified.
  - 19 instances have reached an operational (or equivalent) stage; no flight failures publicly attributed to cFS were identified.
- "Families" of similar OS SW and HW instances limit the total novelty between all instances. These families demonstrate potential for constraining V&V costs through substantial reuse of V&V assurance evidence. The Lunar Reconnaissance Orbiter (LRO) (2009), Radiation Belt Storm Probes (RBSP) / Van Allen Probes (2012), and Global Precipitation Measurement (GPM) (2014) satellites shared common elements that suggest the limit of commonality within a family.
  - The identification of HW/SW families does not appear to reflect a trend to consolidate HW/SW configurations. The expanding range of project software

[7] "Use of the Capability Maturity Model Integration (CMMI Registered Trademark) in Software Engineering Management on NASA Missions" Tim Crumbley, NASA Deputy Manager for Software Engineering Technical Discipline Team, https://ntrs.nasa.gov/citations/20160006992

[8] : *"…most organizations estimated the percentage of time on software testing to be between 30 to 50 percent of the development life cycle. A rule of thumb used by one organization was to plan to test twice as long as you code…"* NASA Software Engineering Benchmarking Study, https://ntrs.nasa.gov/citations/20130013477

[9] *"…testing approach is that the software test teams do not repeat previous tests done by the COTS or GOTS developer, but instead run a comprehensive set of tests in the context of the full build of the project's application of the COTS and GOTS Software…"* NASA Software Engineering Benchmarking Study, https://ntrs.nasa.gov/citations/20130013477

[10] "Project" will be generically used in this document rather than the longer "project or program," but project should be understood to reflect both project cFS use and program cFS use.

classes (driving V&V requirements), operating systems, processor hardware, and evolving Agency leadership direction (driving new requirements or eliminating prior instructions in NPR 7150.2) supports the user communities' application of the abstractions engineered into cFS. It is reasonable to expect that users will continue to apply the capability that has been delivered.
    - These projects include a substantial variety of hardware, software operating systems, and software applications with a generally consistent cFS middle ware. Modifications/optimizations and recommendations for changes in the cFS related code have been identified by the cFS user community.
- Shared characteristics of flight software have not resulted in transferable software assurance evidence while surveyed projects have limited cFS V&V in response to perceptions of lowered risk and increased code maturity.
- Evidence used within assurance or certification efforts could be transferable (e.g., hardware certification by analysis using similarity data), but this approach has not been observed within the NASA cFS software community.
- SCA of the open source cFS related software suggests that with the exception of string processing, the cFS software is generally free of the kinds of potential bugs that SCA readily identifies. The source code generally holds to standards that lead to understandable and maintainable software while string processing related warnings suggest individual targets for code improvements.
    - Static code analysis results were observed to be among the most readily transferred verification and validation evidence. NASA IV&V has a standing tradition not to repeat SCA when applying the same tools to the same code.
- Projects contacted have confidence in the cFS code and perceive it to be both stable and mature.
- Multiple SW projects were not explicitly verifying their cFS SW instance, instead performing validation of the cFS through incorporation of the cFS within the larger system during mission-specific application verification testing and verification testing at the subsystem or high level of assembly.
    - This approach has been accepted (e.g., for the NASA Orion Program) by the appropriate NASA Technical Authority (TA).
    - The gap between the accepted approach and the written guidance document was not anticipated by the OSS cFS FSW V&V project team, and analysis activities were re-planned to accommodate the loss of those data sources.
    - The binary evaluation of accepting the risk of flying incompletely V&Ved cFS software or spending money to V&V software perceived as low risk, omits a third option of developing reusable cFS V&V evidence and identifying V&V gaps that each project should address individually.
- Class "B" cFS reuse and expansion of the user community to Class "A" safety-critical SW is at least partially based on the perception of cFS success within Class "B" uses rather than detailed consideration of cFS assurance evidence, while others, e.g., Johnson Space Center (JSC) based Advanced Exploration Systems (AES) engineers, appear to be attempting nearly comprehensive verification and validation of their cFS instance.

Project/Program	Operating System (If Reported)	Hardware(If Identified)	Launch (If Known)
Lunar Reconnaissance Orbiter (LRO)	VxWorks	RAD750 (PowerPC 750 family)	2009
Morpheus	VxWorks	AiTech S950 (PPC750FX PowerPC 750 family)	2011
Radiation Belt Storm Probes (RBSP) / Van Allen Probes	VxWorks	RAD750 (PowerPC 750 family)	2012
Lunar Atmosphere and Dust Environment Explorer (LADEE)	VxWorks	Unk	2013
Global Precipitation Measurement (GPM) mission	VxWorks	RAD750 (PowerPC 750 family)	2014
Observatory for Planetary Investigations from the Stratosphere (OPIS)	Xenomai Linux	Intel Duo	2014
Magnetospheric Multiscale Mission (MMS)	RTEMS	Rad Hard Coldfire (5208)	2015
Dellingr	FreeRTOS	Gomspace Nanomind A712d ARM7 RISC processo	2017
Neutron star Interior Composition Explorer (NICER)	VXWorks	BRE440 PowerPC	2017
Simulation-to-Flight 1 (STF-1)	FreeRTOS	Gomspace Nanomind A3200 AVR3200 MCU	2018
Compact Radiation Belt Explorer	Linux	XB1 Bus with Cubesat/Chrec Space Processor Inst	2018
Parker Solar Probe (PSP)	RTEMS	LEON3 UT699	2018
Global Ecosystem Dynamics Investigation (at ISS) (GEDI)	VxWorks	PowerPC 440	2018
Seeker ISS Flight Experiment	Linux	CHREC space processor	2019
Coordinated Applied Capitol Technology University Satellite (CACTUS-1)	Unk	Unk	2019
Kenobi ISS Flight Experiment	Unk	Unk	2019
Orion Ascent Abort 2 (AA-2)	VxWorks	AiTech SP0 1 GHz SBC PowerQUICC III processor	2019
Space Test Program -Houston 6, USAF-NASA Goddard	Unk	Unk	2019
Int-Ball2 at ISS JEM	Linux	JETSON TX2 NVDIDIA Pascal	2020
Orion Camera Controller Artemis I (EM-1)	Linux	Intel i5 CPU (NUC)	2021
Lunar IceCube	Linux	P400	2021
MX-1, MX-2, MX-5, MX-9 (CLPS) Moon Express	Unk	Unk	2021
Peregrine Lander - Commercial Lunar Payload Services (CLPS) Astrobotic Technology	Unk	Unk	2021
BioSentinel	VxWorks	UT700 LEON 3FT	2021
Orion Vision Processing Unit Artemis I (EM-1)	VxWorks	UT700 LEON 3FT	2021
SkyFire	Unk	Unk	2021
XL-1 Lander (CLPS) Masten Space Systems, Inc.	Unk	Unk	2021
Orion Camera Controller Artemis II (EM-2)	Linux	Intel i5 CPU (NUC)	2022
Plankton, Aerosol, Cloud, ocean, Ecosystem (PACE)	VxWorks	MUSTANG (custom LEON3 Dual core + LEON3-FT i	2022
Orion Vision Processing Unit Artemis II (EM-2)	VxWorks	UT700 LEON 3FT	2022
Lunar Gateway - Minimal Habitation Module (formerly Utilization Module)	VxWorks	Unk	2023
Exploration Extra-vehicular Mobility Unit (xEMU) Caution & Warning System (CWS)	VxWorks	Leon3 SPARC processor	2023
Roman Space Telescope (RST) (previously Wide-Field Infrared Survey Telescope - WFIRST)	RTEMS	Custom LEON4	2025
Mars Ascent Vehicle (preliminary)	Not App.	Sphinx	
Ames Modular Common Spacecraft Bus	VxWorks	RAD750 processor, 1GB TMR NVRAM	
Avionics & Software Platform for Exploration Capabilities & Technologies (ASPECT)	VxWorks	SP0 processor PowerQUICC III processor	
Certification of cFE on VxWorks ARINC-653	VxWorks ARINC 653	SP0 processor PowerQUICC III processor	

Table 1 – Projects and Programs Applying cFS Code

- There are indicia of unresolved problem reports being managed by the cFS project, but not necessarily considered by cFS's potential future users. Agency guiding documents include the expectation for OTS software users to periodically review OTS FSW defects for possible project impacts[11].

Results were briefed to the NASA community via a Webinar sponsored by the NASA Safety Center (NSC) and held on June 17, 2020.

[11] Reference NPR 7150.2 sub f "The *project manager shall satisfy the following conditions when a COTS, GOTS, MOTS, or reused software component is acquired or used: … f. The project has a plan to perform periodic assessments of vendor reported defects to ensure the defects do not impact the selected software components.*

Final Status

Review and analysis of the V&V of NASA's open source cFS within high-value NASA programs: This goal was modified as a result of Orion (the primary target of concurrent SARP and NASA IV&V project analysis) moving away from formal verification and validation of the cFS software and the acceptance of this position by the Technical Authority. The TA has also communicated his expectation that future instances will not be expected to apply SWE-027 sub e literally but instead to achieve equivalent flight readiness.

Identification of reusable program or project cFS V&V elements or assurance evidence: Projects are explicitly reusing, and reusing with modification, cFS software V&V tools including requirements, success criteria, and test scripts. The reuse of cFS V&V evidence was not observed except that prior verification and validation success was known, and this has decreased the demand for verifying new instances of cFS software (a holistic rather than detailed version of assurance evidence reuse). Process reuse is generally applicable. Tool use is very frequently possible. Limited potential for evidence reuse has been demonstrated; see Figure 2 – Potential cFS V&V Assurance Reuse.

Identification of abstraction levels with potential to limit SW assurance effort: The elimination of the Orion Program as a source of OSS cFS FSW V&V data has substantially limited the objective evidence in this area. The OSS cFS FSW V&V project team observed in conversation with targeted projects that the HW and OS abstractions have been successful. It has also been observed (e.g., in the Orion Program's use of cFS SW) that projects are making direct calls to OS and HW layers, undermining the use of abstractions and creating OS and HW specific instances.

Establish a template for tailoring cFS V&V using the NASA IV&V verification and validation lifecycle as a baseline: This material was completed and attached to this report as Appendix B. "cFS Verification and Validation Reuse".

Phase	Activity	High Level Process Reuse	Tool Reuse	cFS Evidence Reuse
Architecture	System and Avionics Architecture	Yes	Partial	None[1]
	Software Architecture	Yes	Partial	Partial
	cFS Architecture	Yes	Yes	Yes
Requirements	Systems Requirements	Yes	Partial	None
	Avionics Requriements	Yes	Partial	Partial
	Software Requirements	Yes	Yes	Partial
	cFS Requirements	Yes	Partial	Partial
Test	Unit Test	Yes	Yes	Yes
	Integrated HW/SW Test	Yes	Partial	None[1]
	CSCI Verification	Yes	Partial	Partial
	CSCI Validation	Yes	Partial	Partial
	Systems Test	Yes	Partial	None
Design	Algorithm Correct and Complete	Yes	Partial	Yes (if fully documented)
	Requirements Trace	Yes	Partial	Yes (if fully documented)
	Interface Design	Yes	Partial	Partial
	Security	Yes	Partial	Partial
Implementation	CSCI Requirements and Design Trace	Yes	Yes	Yes (if fully documented)
	CSCI Algorithm Trace	Yes	Yes	Yes (if fully documented)
	Interface Implementation	Yes	Partial	Partial
	Security	Yes	Partial	Partial
	CSCI Code Quality	Yes	Yes	Yes

1) Instances of HW/SW "Family" reuse within similar projects with similar processing and safety constraints provide evidence in those cases. Establishing that such limitations are met is novel analysis and therefore "cFS Evidence Reuse" was scored "None".

Figure 2 – Potential cFS V&V Assurance Reuse

Documentation of multiple projects' verification and validation approach: Completed for the announced analysis targets.

Documentation of verification and validation tool reuse: Completed for announced analysis targets.

Klocwork static code analysis of cFS code as a gauge of likely code maturity and dependability including delivery of higher impact SCA issues to cFS team: Analysis is complete. Candidate issues were delivered to the cFS development project in Fiscal Year (FY) 2019.

The SARP Open Source cFS FSW V&V project identified a candidate risk to NASA flight assurance goals from external user's perception of cFS maturity, stability, and low risk driving

cFS acceptance without performing targeted verification and validation. This candidate risk, included in Appendix C., has been provided to the NASA Technical Fellow for Software Assurance, Office of Safety and Mission Assurance (OSMA) for consideration as an agency-level risk.

The SARP project also identified four potential candidate risks associated with projects' use of cFS middleware that will have various scores for risk consequence and risk likelihood based on project-specific evaluation. Guidance for establishing the cFS middleware risk consequence and likelihood is provided in Appendix D.. These candidate risks, unlike the agency-level candidate risk mentioned above, potentially apply to each project using cFS middleware and may have different risk consequence and risk likelihood scores from project to project.

1 Introduction

The core Flight System (cFS) is based on Goddard Space Flight Center (GSFC) heritage NASA Class "B" software and was initially applied to Class "B" and other less critical flight uses. The cFS software has expanded its NASA mission footprint into both additional science missions and safety-critical human-rated flight software (FSW) architectures. The expansion of the footprint changes (i) the NASA Centers using and assuring cFS based systems (moving outside the teams with previous cFS experience), (ii) system fault impacts (shorter times to criticality or more critical fault impact), and (iii) mission types (crew vehicles, crewed systems). The expansion of cFS use represents opportunities for improvements in cost, schedule, and technical excellence, as well as a source of risks arising from moving from prior (largely successful, but higher cost) FSW approaches to a new use of this demonstrated approach within a different environment.

The cFS architecture and framework approach to FSW is described by its NASA development team as *"a platform and project independent reusable software framework and set of reusable software applications"… "suitable for reuse on any number of NASA flight projects and/or embedded software systems at a significant cost savings".* The goals of the cFS FSW approach have been represented in slightly different ways in various different forums. The goals presented below in Table 2 are identical to some other citations and generally similar to all representations of the cFS goals or announced achievement[12].

	Goal
1	Reduce time to deploy high quality flight software
2	Reduce project schedule and cost uncertainty
3	Directly facilitate formalized software reuse
4	Enable collaboration across organizations
5	Simplify sustaining engineering (AKA. On Orbit FSW maintenance)
6	Scale from small instruments to Hubble class missions
7	Build a platform for advanced concepts and prototyping
8	Create common standards and tools across the center

Table 2 – Representative cFS Approach Goals[13]

It is noteworthy that

- Software Assurance (SA) including Verification and Validation (V&V) is not addressed within the expressed cFS Approach Goals.[14]
- cFS has migrated into Class "A" and safety-critical applications while use in Class "A" FSW systems was not initially forecasted.

[12] *"The cFS architecture has been proven to: Reduce time to deploy high quality flight software, Reduce project schedule and cost uncertainty, Facilitate formalized software reuse, Enable collaboration across organizations, Simplify flight software sustaining engineering, Provide a platform for advanced concepts and prototyping, Provide common standards and tools across Goddard's missions and NASA wide"* https://cfs.gsfc.nasa.gov/

[13] https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20140017040.pdf

[14] Software assurance might be an element of achieving goals 1, 2, 3, and 6.

Each cFS project uses a unique instance and configuration of hardware (HW) and software (SW) including: (i) Commercial Off The Shelf (COTS) operating systems, (ii) cFS components and versions, (iii) reusable application programs, and (iv) mission-specific application programs.

1.1 Purpose

The Software Assurance Research Program (SARP) sponsored Open Source Core Flight System Flight Software Verification & Validation project reviewed and analyzed V&V of NASA's open source software (OSS) cFS within high-value NASA programs to establish the potential for V&V software assurance evidence reuse. The reuse of cFS software represents a demonstrated approach for achieving developer cost and schedule savings as well as improved cost and schedule certainty. Establishing a mechanism for performing rigorous software assurance evidence reuse would establish objective satisfaction of rigorous V&V requirements and cost and schedule savings.

Multiple NASA Projects or Programs have applied or are currently applying cFS SW, including recent expansion into safety-critical software NASA Class "A" software. Public statements by NASA, international partners, and other aerospace actors have assisted this project in identifying 41 previous or current cFS instances. These projects or programs and the anticipated first year of operation[15] are displayed in "Table 3 – Identified Projects or Programs with Associated Launch Date if Applicable".

It is important to note that acceptable SW risk varies across projects and software instances become certified. Software is not certified generally. The relatively large number (for a NASA SW project) of cFS users across the identified programs with shared software characteristics has not resulted in transferable software assurance evidence or a common flight certification.

The appropriate degree of software assurance rigor can be related to the risk of software failure, e.g.: (a) increasing rigor within applicable Software Engineering requirements of NASA Procedural Requirements (NPR) 7150.2 as the software Class increases from D, to C, to B, to A.[16], and (b) increasing expectations within NASA-STD-8719.13 "NASA Software Safety Standard" for safety-critical software versus other software.[17]

[15] Launch date or equivalent for ground-based projects.

[16] Reference NPR 7150.2 "Appendix C. Requirements Mapping and Compliance Matrix"

[17] Reference SSS-002 of NASA-STD-8719.13 *"When safety critical software is developed or acquired by or for NASA, the acquirer organization shall meet the requirements of this Standard".*

	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
Avionics & Software Platform for Exploration Capabilities & Technologies (ASPECT)																					X	
Ames Modular Common Spacecraft Bus																					X	
Mars Ascent Vehicle (preliminary)																				X		
CATALYST (3 partners) 2014																				X		
Certification of cFE on VxWorks ARINC-653																				X		
Roman Space Telescope (RST) (formerly the Wide-Field Infrared Survey Telescope (WFIRST)																				X		
Lunar Gateway - Minimal Habitation Module (formerly Utilization Module)																				X		
Exploration Extra-vehicular Mobility Unit (xEMU) Caution & Warning System (CWS)																				X		
Orion Vision Processing Unit Artemis II (EM-2)																			X			
Orion Camera Controller Artemis II (EM-2)																			X			
Plankton, Aerosol, Cloud, ocean, Ecosystem (PACE)																			X			
Lunar Gateway																			X			
Peregrine Lander - Commercial Lunar Payload Services (CLPS) Astrobotic Technology																			X			
XL-1 Lander (CLPS) Masten Space Systems, Inc.																			X			
MX-1, MX-2, MX-5, MX-9 (CLPS) Moon Express																			X			
BioSentinel																			X			
Orion Vision Processing Unit Artemis I (EM-1)																			X			
Orion Camera Controller Artemis I (EM-1)																			X			
Lunar IceCube																			X			
SkyFire																			X			
Int-Ball2																		X				
Orion Ascent Abort 2 (AA-2)																	X					
Coordinated Applied Capitol Technology University Satellite (CACTUS-1)																	X					
Space Test Program -Houston 6, USAF-NASA Goddard																	X					
Seeker ISS Flight Experiment																	X					
Kenobi ISS Flight Experiment																	X					
Simulation-to-Flight 1 (STF-1)																X						
Compact Radiation Belt Explorer																X						
Global Ecosystem Dynamics Investigation (at ISS) (GEDI)																X						
Parker Solar Probe (PSP)																X						
Dellingr															X							
Neutron star Interior Composition Explorer (NICER)															X							
Magnetospheric Multiscale Mission (MMS)													X									
Orion Exploration Flight Test 1 (EFT-1)												X										
Observatory for Planetary Investigations from the Stratosphere (OPIS)												X										
Global Precipitation Measurement (GPM) mission												X										
Lunar Atmosphere and Dust Environment Explorer (LADEE)											X											
Radiation Belt Storm Probes (RBSP) / Van Allen Probes										X												
Morpheus									X													
Lunar Reconnaissance Orbiter (LRO)							X															
CHIPSat (Precursor)	X																					

Table 3 – Identified Projects or Programs with Associated Launch Date if Applicable

Previously observed instances of cFS reuse include the FSW developers' modification and reuse of legacy SW verification test processes. Observed instances of NASA's reuse of open source cFS has not reflected a significant reuse of software assurance evidence. Thus, cFS reuse had not been witnessed to contribute directly to software assurance cost and schedule savings while meeting NASA requirements for SW V&V. NASA cFS users may not be effectively capturing and communicating the verification and validation results from prior uses of the cFS software resulting in either unnecessary risks (e.g., incomplete V&V, incomplete

application of "lessons learned", overstatement of prior V&V applicability to current missions) or unnecessary V&V costs (e.g., repetition of V&V activities that do not substantially aid mission assurance).

The identified record for the intended benefits of the cFS approach has not identified verification and validation reduction or V&V cost reduction as an intended benefit. At the time of the cFS's development and propagation, the high-level NASA expectations such as those found within NASA NPR-7150.2 "Software Engineering Requirements" included a requirement for off-the-shelf (OTS) software to be verified and validated to the same "level" or "level of confidence" as developed software[18]. This requirement should restrict the degree to which cFS reuse would result in V&V cost savings. The ability to reuse SA V&V evidence would provide a rigorous and flexible match for the high-level NASA leadership expectation for OTS software V&V equal to developed code with cost savings and improved scheduled certainty.

The continued use of cFS and the absence of a documented strategy to reduce V&V cost, schedule, cost risk, and schedule risk represents an opportunity for improvements.

1.2 Project Summary and System Overview

1.2.1 Description of the cFS

The cFS architecture is based on multiple layers of abstractions intended to limit the impact of hardware or software changes while supporting the insertion of mission-specific applications, thus facilitating meaningful formalized software reuse with concomitant improvements in software cost, schedule and quality. A representation of this layered architecture is presented in Figure 3 below.

[18] Reference NPR 7150.2 SWE-027 Rev I/R "*Off-the-shelf software is validated to the same level of confidence as would be required of the developed software."* or Rev A, Rev B and Rev C *"The software component is verified and validated to the same level required to accept a similar developed software component for its intended use."*

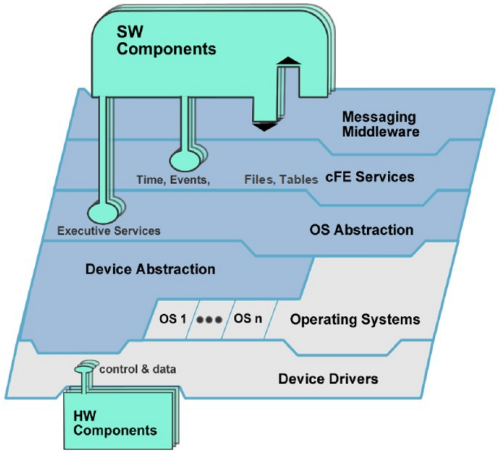


Figure 3 – cFS Layered Service Architecture[19]

In addition to the incorporation of the explicitly reusable device abstraction, OS abstraction, Core Flight Executive (CFE) services, and messaging middleware, the cFS architecture will

- permit a growing library of reusable or partially reusable mission applications tailored to individual user's project needs, and
- establish defacto internal interface practices that facilitate rapid application development reusable across missions and projects.

Omitted from is a graphical representation of a limitation of the abstractions. E.g., direct calls to the OS from application SW is possible and was observed within projects reviewed. Thus, portability is supported but not assured by the abstraction architecture.

An alternative representation of the cFS FSW Layers is provided in . This representation introduces the sources for application software. The theoretical ability to reuse application software verification and validation evidence would be dependent on both the reuse of the application software and the reuse of interfacing software and hardware.

[19] National Aeronautics and Space Administration https://cfs.gsfc.nasa.gov/cFS-OviewBGSlideDeck-ExportControl-Final.pdf

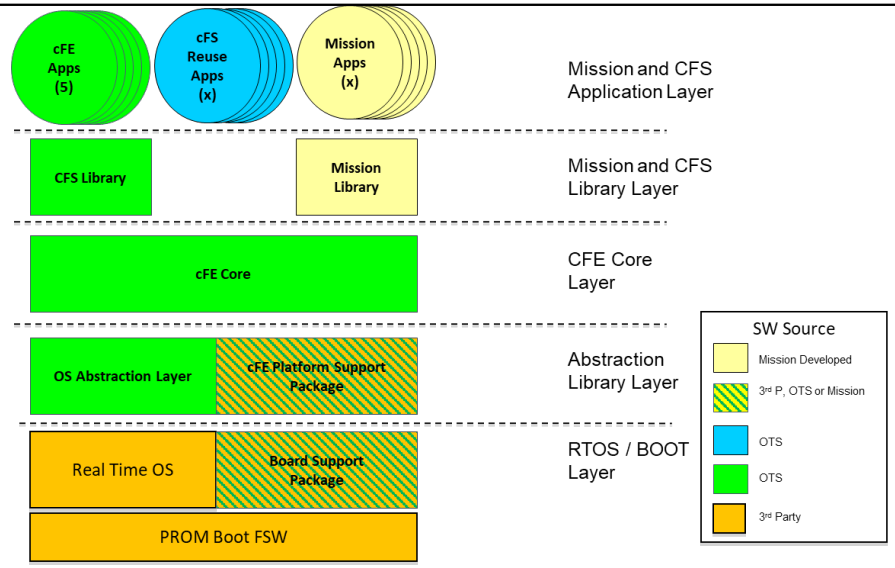


Figure 4 – cFS Flight Software Layers[20]

The cFS configurable set of applications and services incorporates approaches to manage growth and limit complexity. A generic representation of a cFS architecture is provided in "Figure 5 – Generic cFS SW Architecture". This type of illustration provides an easy method of communicating both general software structure (excluding the operating system) and identifying new development and reused application software. Visible in the diagram is a simple representation of the cFS software bus used to limit development complexity and manage growth. The software bus is a publish and subscribe service to simplify the interfaces between applications or between cFS services and applications. Also represented within Figure 5 are (a) program/project developed (i) Input/Output (IO) applications and (ii) mission applications, (b) configurable cFS applications, and (c) cFS services.

The NASA cFS community provides reusable software artifacts including (a) SW requirements, (b) design documentation, (c) test procedures, (d) test results, (e) development standards, and (f) user guides.

[20] cFE_CFS_Overview_Cudmore_FSW2011 Slide 17 cFS17-B1-Workshop_Platforms-Cudmore Slide 9 and 10

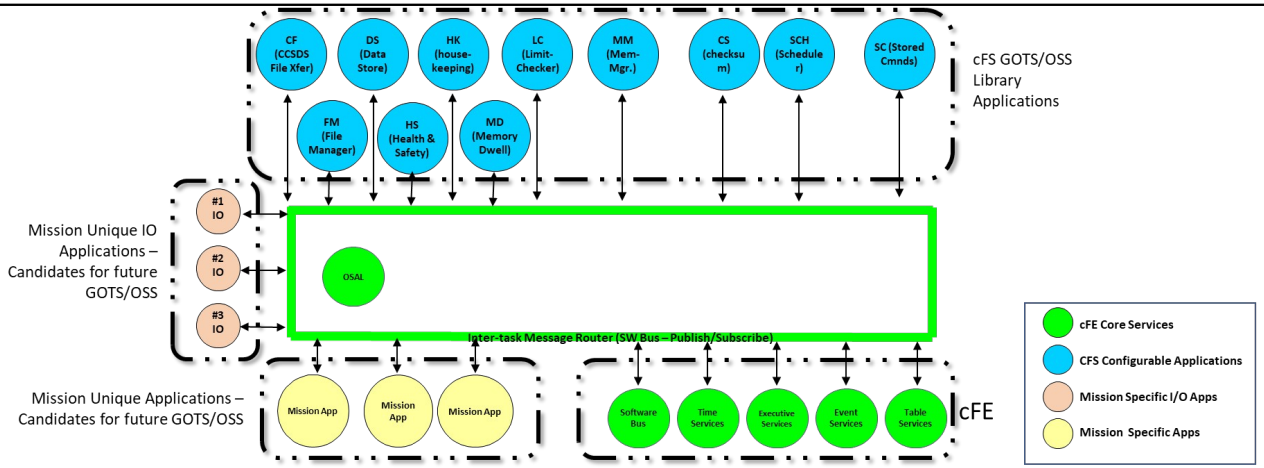


Figure 5 – Generic cFS SW Architecture

The cFS community claims a number of potential SW testing advantages including: (a) the potential to run/test SW applications on both desktop systems and embedded target system without SW changes, (b) the potential for early and continuous SW testing starting before the embedded target system HW is available, (c) the potential for test suite reuse, and (d) the potential for test procedures/results reuse.

1.3 OSS cFS FSW V&V SARP Analysis Goals

The goals for the OSS cFS FSW V&V SARP research project included: (a) the review and analysis of the V&V of NASA's open source core Flight System (cFS) within high-value NASA programs; (b) the identification of reusable program or project cFS V&V elements or assurance evidence; (c) identification of abstraction levels with potential to limit software (SW) assurance effort; and (d) establishment of a template for tailoring cFS V&V using the NASA Independent Verification and Validation (IV&V) verification and validation lifecycle as a baseline.

The OSS cFS FSW V&V SARP research project was also designed to provide incidental assurance to current cFS projects and programs including the three current Orion Multi-Purpose Crew Vehicle (MPCV) instances of cFS: (i) Ascent Abort-2 (AA-2) flight test Crew Module flight computer, (ii) Artemis x Vision Processing Unit (the dissimilar FSW processor for Orion MPCV crewed flight), and (iii) Camera Controller (supporting Artemis x flight Optical Navigation and non-critical image capture). The completion of the Klocwork static code analysis (SCA) is one element of this planned incidental assurance.

2 NASA IV&V Analysis Summary

To accomplish the objectives described in the previous section, the NASA IV&V team evaluated project artifacts using defined analysis methods, targeting the applicable OSS capabilities, and providing coverage across development phases, including requirements, implementation, and test.

2.1 Static Code Analysis

Before performing an assessment on the cFS/cFE source code, the OSS cFS FSW V&V project team obtained the GSFC-developed open source code from SourceForge. The files downloaded showed they were mature in that they had not been updated recently. The downloads consisted of the software identified in Table 4 below.

cFE-6.5.0-OSS-release	cfs-lc-2.1.0-OSS-release
cfs-cf-2.2.1-OSS-release	cfs-md-2.3.1-OSS-release
cfs-cs-2.4.0-OSS-release	cfs-mm-2.4.1-OSS-release
cfs-ds-2.5.1-OSS-release	cfs-sbn-1.0.0-OSS-release
cfs-fm-2.5.2-OSS-release	cfs-sc-2.5.0-OSS-release
cfs-hk-2.4.1-OSS-release	cfs-sch-2.2.1-OSS-release
cfs-hs-2.3.0-OSS-release	osal-4.2.1a-release

Table 4 – Software Subjected to Static Code Analysis

The software analysis tool Klocwork was run on all of the software once it had been extracted. This tool was chosen since it has better results for coding standards than the other tools in NASA IV&V's possession at the time as well as having good checkers for logic errors. The process of getting warnings to analyze was:

1. Run the Klocwork analysis tool on the software
2. Extract the warnings from the tool in a form compatible with Microsoft Excel
3. Use Excel functions to identify tooling and test files which were ignored

The tool found 20,448 warnings in the FSW files.

NASA IV&V recognizes four sets of warnings generated by SCA tools, which are referred to as "categories". Category 1 warnings tend to generate high impact issues. Examples are null pointer dereferences and array bounds violations. Category 2 warnings tend to have high false positive rates and can be difficult to prove as true positives. Examples are casting pointer to structs to pointer of

different size structs, and precision loss in calculations. Since these errors tend not to cause subtle effects, testing is a cost effective way to find them. Category 3 warnings are coding standard violations that affect readability and maintenance but not directly the functioning software. Examples are requiring all switch statements to have a *default* case and forbidding functions with variable length argument lists. Category 3 warnings tend to have high true positive rates. The final category, Category 4, is also coding standards but for which there is division in the community about the appropriateness of the standard. Examples are forbidding the use of the ternary operator and requiring all operands to logical operators to be of Boolean type. Other warnings in the final category are those that are never found to generate issues or that if true would prevent compilation of the source. For this analysis, only the Category 1 warnings were analyzed in detail.

Once the warnings from test files and utilities and Category 4 were ignored, there were 12,646 warnings resulting from flight software. The distribution of the warnings across categories and cFS applications is detailed in Table 5 below.

Software Application		Category				Category by Percent		
		1	2	3	Total	1	2	3
CFDP File	**CF**	90	1952	1155	**3197**	2.82%	61.06%	36.13%
core Flight Executive	**cFE**	172	3692	679	**4543**	3.79%	81.27%	14.95%
Checksum	**CS**	8	489	45	**542**	1.48%	90.22%	8.30%
Data Storage	**DS**	16	405	42	**463**	3.46%	87.47%	9.07%
File Manager	**FM**	15	441	62	**518**	2.90%	85.14%	11.97%
Housekeeping	**HK**	3	70	5	**78**	3.85%	89.74%	6.41%
Health and Safety	**HS**	9	183	12	**204**	4.41%	89.71%	5.88%
includes		42	2	86	**130**	32.31%	1.54%	66.15%
Limit Checker	**LC**	0	214	43	**257**	0.00%	83.27%	16.73%
Memory Dwell	**MD**	12	194	21	**227**	5.29%	85.46%	9.25%
Memory Manager	**MM**	17	507	83	**607**	2.80%	83.53%	13.67%
OS Abstraction Layer	**osal**	57	803	97	**957**	5.96%	83.91%	10.14%

Software Bus Network	**SBN**	33	213	65	**311**	10.61%	68.49%	20.90%
Stored Commanding	**SC**	6	378	57	**441**	1.36%	85.71%	12.93%
Scheduler	**SCH**	2	156	13	**171**	1.17%	91.23%	7.60%
Grand Total		**482**	**9699**	**2465**	**12646**	**3.81%**	**76.70%**	**19.49%**

Table 5 - Distribution of SCA Warnings across Categories and cFS Applications

The table shows that most of the warnings are Category 2 warnings, which is not unexpected in the systems-like programming environment of FSW development. Most applications have between 1 and 6 percent of the warnings as Category 1 warnings. The three outliers are includes, Limit Checker, and Software Bus Network. Limit Checker had no Category 1 warnings and had no issues resulting from this analysis. "includes" is not a cFS application but is an artifact of NASA IV&V's use of the Klocwork tool; it contains all of the include files in one location. Software Bus Network had a high percentage of Category 1 warnings but ended up with few issues from them.

The source code indicated for each Category 1 warning was analyzed in detail to determine if there was a Potential Bug (PB) issue, a Coding Standard (CS) issue, or a False Positive (FP). Most warnings are false positives. Table 6 below shows the issues resulting from each application.

Software Application		Issues Identified				IVV severity by percent		
		PB	CS	FP	Total	PB	CS	FP
CFDP File	**CF**	13	9	68	90	14%	10%	76%
core Flight Executive	**cFE**	16	10	146	172	9%	6%	85%
Checksum	**CS**	5	2	1	8	63%	25%	13%
Data Storage	**DS**	2	2	12	16	13%	13%	75%
File Manager	**FM**	9	0	6	15	60%	0%	40%
Housekeeping	**HK**	1	0	2	3	33%	0%	67%
Health and Safety	**HS**	4	1	4	9	44%	11%	44%

includes		0	1	41	42	0%	2%	98%
Limit Checker	**LC**	0	0	0	0	NA	NA	NA
Memory Dwell	**MD**	5	3	4	12	42%	25%	33%
Memory Manager	**MM**	11	1	5	17	65%	6%	29%
OS Abstraction Layer	**osal**	14	1	42	57	25%	2%	74%
Software Bus Network	**SBN**	4	1	28	33	12%	3%	85%
Stored Commanding	**SC**	0	0	6	6	0%	0%	100%
Scheduler	**SCH**	0	0	2	2	0%	0%	100%
All		**84**	**31**	**367**	**482**	**17%**	**6%**	**76%**

Table 6 – Issues Resulting from Each cFS Application

Three applications did not have any potential bugs identified through this process and two more had two or fewer each.

Most of the potential bugs involve string processing through either reading a file and not parsing field sizes or using concatenation to build fully qualified file names. In both cases, overflowing the destination buffer is possible. The former case is reading outside user-supplied data, allowing an ill-informed or malicious user to create problems.

There is also a set of potential bugs resulting from the use of strncpy where the maximum number of characters to copy is the size of the destination buffer. This does not leave room for the null character to be placed at the end of the string. This would result in faulty processing of the string later with unanticipated effects. This is a kind of bug that could go unnoticed for a period of time depending on the data adjacent to the destination buffer.

With the exception of string processing, the cFS software is generally free of the kinds of potential bugs that SCA would be capable of finding.

Performing the analysis of the SCA warnings gave the analyst the opportunity to read segments of the software from a broad swath of the cFS source code. The general impression is that the source code is clean, clear and well documented. Public interfaces are documented with Doxygen comments that appear to follow the standards as stated in the cFS Development Standards version 1.2. The implementation source code is well documented with concise comments that describe what the code is doing without simply restating the source code itself. The analyst found that the source code was generally less convoluted than other flight software.

The Klocwork SCA tool produced 2,465 Category 3 warnings which reflect adherence to coding standards. Of those, 952 were produced where the consequent statement of an *if* statement was not a compound statement; no curly braces were used. With the size of the source code base, NASA IV&V would have expected more warnings and more serious warnings.

The cFS software was held to the GSFC Flight Software Branch – Code 582 C Coding Standard document. The only standard found in the warnings that was specified in the standard document was about using break statements for *case* statements in *switch* blocks. There were eight of these. That standards document offers advice about statement structure but is sparse with respect to "shall" statements. The OSS cFS FSW V&V project team recommends that developers developing applications for cFS refer to the Software Engineering Institute (SEI) C coding standards.

The OSS cFS FSW V&V project team observed two practices that might warrant creating standards to avoid. First, there was a consistent practice of obtaining the address of an array by taking the address of the first element in the array. Instead of using the name of the array as the address (e.g., myArray), the practice was to get the first element and take the address of it (e.g., &myArray[0]). Both methods yield the same value, but the first has the type of address of array and the second has the type address of an element. Though generally interchangeable, this difference in type can cause subtle problems.

Second, there were a couple of cases where a macro was used to rename labels in an enumeration. This indirection hides the fact that an enumeration is being used and could lead to maintenance problems.

There was at least one file that did not strictly adhere to a consistent indentation pattern.

This analysis showed that the source code generally held to standards that should lead to understandable and maintainable software.

2.2 Design and Implementation Analyses

The OSS cFS FSW V&V project team intended to perform design and implementation analysis to assure that the software requirements for each application flowed correctly to design and source code. Included in each app package was a set of presentations labeled 'Design Review,' which provided a summary of the app, including various diagrams, functions, commands, and other useful context. However, the design materials available did not support valuable V&V analysis below the level of architecture.

The design presentations, taken in combination with application requirements and version description document, provided the necessary context to begin source code analysis. The analysis involved manually creating the requirements-to-code traces because the Open Source cFS FSW V&V project team did not identify a traceability matrix between requirements, design, and source code. Using the design presentations, requirements, and unit test cases as a reference, the team manually created a requirements-to-code mapping for each of the apps analyzed and made comments on identified weaknesses or deficiencies. Analyses established source code for most requirements. Some instances could not be identified. Some implementation was found to implement the code incorrectly.

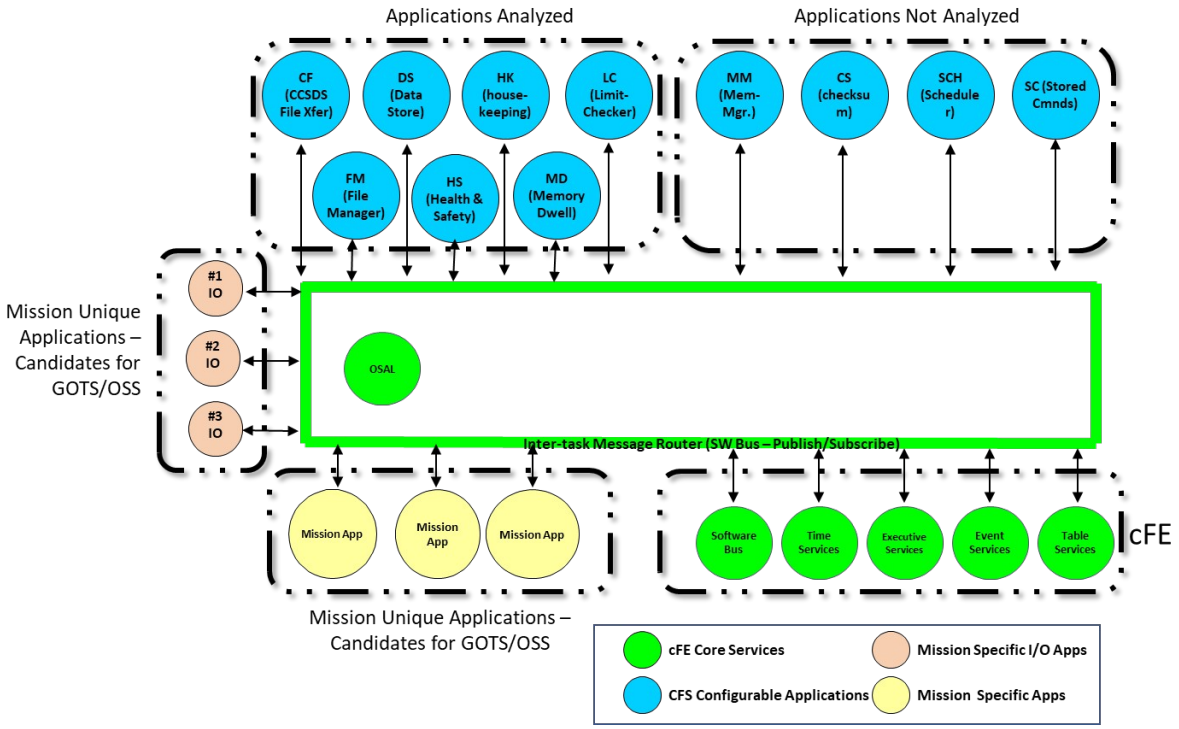


Figure 6 – Targeted and Untargeted cFS Software

The following characterizations were developed:

- Design documentation did not meet leadership's expectations expressed in NPR 7150.2 SWE-052.[21] - NASA Software Engineering Requirements SWE-052 requires bi-directional traceability between (i) software requirements and software design components and (ii) software design components and source code. The NASA Software Engineering Handbook NASA-HDBK-2203 identifies the requirement for the trace to the "software component" as defining a trace to low-level design. These traces were considered gaps in the V&V information delivered for reviewed applications.
    - The independently generated trace was a useful exercise. The developer's trace would provide clear picture of traceability between requirements, source code, and unit test.

[21] "*The project manager shall perform, record, and maintain bi-directional traceability between the following software elements: [SWE-052]"* (for Class A, B, and C SW) *..."Higher-level requirements to the software requirements," ..."Software requirements to the system hazards ... Software requirements to the software design components" ... "Software design components to the software code" ... "Software requirements to the software test procedures" ... "Software requirements to the software non-conformances" ...*

  - Traces to source code was complicated by the absence of the traces to the design. The traceability matrix should be included alongside the other references. The developer's trace data facilitates cFS understanding in developers, users, and software assurance engineers.
- Full requirement implementation is unclear – Across cFS apps analyzed, several requirements are heavily compounded, with lists in excess of 10 sub-items within the text itself. In addition to making traceability difficult, there were instances where one or two steps appeared to be missing or omitted in the code. A traceability exercise would be useful to ensure that requirements are correct and that source code is not missing steps or features.
- Source code comments – As either an alternative or supplement to creating a traceability matrix, the cFS source code would benefit from additional comments within the .c files themselves (perhaps including notes about where requirements are being fulfilled). Although the references are helpful, creating manual traceability is time-consuming.

2.3 Reuse of cFS Verification and Validation Lifecycle Evidence

The potential for reusing cFS V&V evidence can be estimated based on currently completed reviews. Verification and validation reuse can be considered in three areas: process reuse, tool reuse, and V&V evidence reuse.

While processes were generally considered reusable and tools could often be reusable, evidence is largely specific to an instance (characterized by the operating system and hardware platform). Observations in this area were captured in Figure 7 – Identified Potential for Reuse of cFS Verification Evidence.

Phase	Activity	High Level Process Reuse	Tool Reuse	cFS Evidence Reuse
Architecture	System and Avionics Architecture	Yes	Partial	None[1]
	Software Architecture	Yes	Partial	Partial
	cFS Architecture	Yes	Yes	Yes
Requirements	Systems Requirements	Yes	Partial	None
	Avionics Requriements	Yes	Partial	Partial
	Software Requirements	Yes	Yes	Partial
	cFS Requirements	Yes	Partial	Partial
Test	Unit Test	Yes	Yes	Yes
	Integrated HW/SW Test	Yes	Partial	None[1]
	CSCI Verification	Yes	Partial	Partial
	CSCI Validation	Yes	Partial	Partial
	Systems Test	Yes	Partial	None
Design	Algorithm Correct and Complete	Yes	Partial	Yes (if fully documented)
	Requirements Trace	Yes	Partial	Yes (if fully documented)
	Interface Design	Yes	Partial	Partial
	Security	Yes	Partial	Partial
Implementation	CSCI Requirements and Design Trace	Yes	Yes	Yes (if fully documented)
	CSCI Algorithm Trace	Yes	Yes	Yes (if fully documented)
	Interface Implementation	Yes	Partial	Partial
	Security	Yes	Partial	Partial
	CSCI Code Quality	Yes	Yes	Yes

1) Instances of HW/SW "Family" reuse within similar projects with similar processing and safety constraints

Figure 7 – Identified Potential for Reuse of cFS Verification Evidence

The *"Core Flight Software Project Sharepoint Site"* [22] associated with the "cFS AES[23] Project" of the Johnson Space Center's (JSC's) Software, Robotics, and Simulation Division of the Engineering Directorate has demonstrated the delivery of the equivalence of instance-specific flight certification content. The AES cFS Project's purpose was stated as *"to evolve and extend the reusability of the cFS system into human rated systems, thus enabling low cost, and rapid access to space."* Instance-specific certification support evidence was developed including: (i) cFS Product Instantiation on Green Hills Aeronautical Radio, Incorporated (ARINC) 653 operating system and Orion HW Platform, (ii) cFS Product Instantiation in Trick Simulation, and (iii) cFS Product Instantiation on VxWorks operating system and LEON3 HW. Additional cFS support was planned including a reusable test suite and performance monitoring tools (examples of tool reuse).

Less formal verification and validation credit has been observed in the prioritization of verification and validation effort. In this less formal approach, the successes and qualities of the prior applications of cFS are considered rather than the specific evidence. Those qualities are then used to establish the need for performing instance-specific V&V (e.g., NASA IV&V's heritage review process or the Orion Program's Artemis I cFS use without instance-specific verification and validation).

3 Reuse of Verification and Validation Evidence

The analyses completed were positive and have established a number of observations:

- There are a significant number of projects or programs[24] successfully applying the cFS code as shown in Table 7 below. cFS use spans a very wide range of software classifications and NASA centers. Substantial variations between cFS instances' operating system (OS) and processing HW have been observed.
  - Adoption of cFS continues; 40 instances have been identified.
  - 19 instances have reached an operational (or equivalent) stage; no flight failures publicly attributed to cFS were identified.
- "Families" of similar OS SW and HW instances limit the total novelty between instances. These families demonstrate potential for constraining V&V costs through substantial reuse of V&V assurance evidence. The Lunar Reconnaissance Orbiter (LRO) (2009), Radiation Belt Storm Probes (RBSP) / Van Allen Probes (2012), and Global Precipitation Measurement (GPM) (2014) satellites shared common elements that suggest the limit of commonality within a family.

[22] https://oasis.jsc.nasa.gov/projects/advdev/CFS/SitePages/Home.aspx?InitialTabId=Ribbon.EditingTools.CPEditTab&VisibilityContext=WSSTabPersistence

[23] Advanced Exploration Systems (AES)

[24] "Project" will be generically used in this document rather than the longer "project or program," but project should be understood to reflect both project cFS use and program cFS use.

  - The identification of HW/SW families does not appear to reflect a trend to consolidate HW/SW configurations. The expanding range of project software classes (driving V&V requirements), operating systems, processor hardware, and evolving Agency leadership direction (driving new requirements or eliminating prior instructions in NPR 7150.2) supports the user communities' application of the abstractions engineered into cFS.
  - These projects include a substantial variety of hardware, software operating systems, and software applications with a generally consistent cFS middle ware.
  - The range of hardware configurations, software operating systems, application reuse, and range of missions addressed suggests the cFS SW achieves a number of the initial goals targeted. It is reasonable to expect that users will continue to apply the capability that has been delivered.
- Shared characteristics of flight software and multiple projects incorporating cFS have not resulted in transferable software assurance evidence while surveyed projects have limited cFS V&V in response to perceptions of lowered risk and increased code maturity.
- Evidence used within assurance or certification efforts could be transferable (e.g., hardware certification by analysis using similarity data), but this approach has not been observed within the NASA cFS software community.
- SCA of the open source cFS related software suggests that with the exception of string processing, the cFS software is generally free of the kinds of potential bugs that SCA readily identifies. The source code generally holds to standards that lead to understandable and maintainable software while string processing related warnings suggest individual targets for code improvements.
  - Static code analysis results were observed to be among the most readily transferred verification and validation evidence. NASA IV&V has a standing tradition not to repeat SCA when applying the same tools to the same code.
- Projects surveyed had confidence in the cFS code and perceived it to be both stable and mature.
- Multiple Class A or B projects are not verifying their instance of cFS. In recent identified cases, the lack of comprehensive cFS verification and validation has been communicated. However, the choice does not appear to be limited to the binary options of (a) do new V&V of the cFS instance or (b) skip instance V&V and accept the associated risk.
  - Multiple SW projects were not explicitly verifying their cFS SW instance nor methodically examining historical records for completed V&V activities where assurance evidence can be reused. Observed projects were instead performing validation of the cFS through incorporation of the cFS within the larger system during mission-specific application verification testing and verification testing at the subsystem or high level of assembly.
  - Some instances of this approach have been accepted (e.g., for the NASA Orion Program) by the appropriate NASA Technical Authority (TA).
  - The gap between the accepted approach and the written guidance document was not anticipated by the OSS cFS FSW V&V project team, and analysis activities were re-planned to accommodate the loss of those data sources.

    - The binary evaluation of accepting the risk of flying incompletely V&Ved cFS software or spending money to V&V software perceived as low risk, omits a third option of developing reusable cFS V&V evidence and identifying V&V gaps that each project should address individually.
        - Cost control/reduction and risk communication may support appropriate conformance to NASA leadership's expectations.
- Class "B" cFS reuse and expansion of the user community to Class "A" safety-critical SW is at least partially based on the perception of cFS success within Class "B" uses rather than detailed consideration of cFS assurance evidence, while others, e.g., JSC-based AES engineers, appear to be attempting nearly comprehensive verification and validation of their cFS instance.

Project/Program	Operating System (If Reported)	Hardware(If Identified)	Launch (If Known)
Lunar Reconnaissance Orbiter (LRO)	VxWorks	RAD750 (PowerPC 750 family)	2009
Morpheus	VxWorks	AiTech S950 (PPC750FX PowerPC 750 family)	2011
Radiation Belt Storm Probes (RBSP) / Van Allen Probes	VxWorks	RAD750 (PowerPC 750 family)	2012
Lunar Atmosphere and Dust Environment Explorer (LADEE)	VxWorks	Unk	2013
Global Precipitation Measurement (GPM) mission	VxWorks	RAD750 (PowerPC 750 family)	2014
Observatory for Planetary Investigations from the Stratosphere (OPIS)	Xenomai Linux	Intel Duo	2014
Magnetospheric Multiscale Mission (MMS)	RTEMS	Rad Hard Coldfire (5208)	2015
Dellingr	FreeRTOS	Gomspace Nanomind A712d ARM7 RISC processo	2017
Neutron star Interior Composition Explorer (NICER)	VXWorks	BRE440 PowerPC	2017
Simulation-to-Flight 1 (STF-1)	FreeRTOS	Gomspace Nanomind A3200 AVR3200 MCU	2018
Compact Radiation Belt Explorer	Linux	XB1 Bus with Cubesat/Chrec Space Processor Inst	2018
Parker Solar Probe (PSP)	RTEMS	LEON3 UT699	2018
Global Ecosystem Dynamics Investigation (at ISS) (GEDI)	VxWorks	PowerPC 440	2018
Seeker ISS Flight Experiment	Linux	CHREC space processor	2019
Coordinated Applied Capitol Technology University Satellite (CACTUS-1)	Unk	Unk	2019
Kenobi ISS Flight Experiment	Unk	Unk	2019
Orion Ascent Abort 2 (AA-2)	VxWorks	AiTech SP0 1 GHz SBC PowerQUICC III processor	2019
Space Test Program -Houston 6, USAF-NASA Goddard	Unk	Unk	2019
Int-Ball2 at ISS JEM	Linux	JETSON TX2 NVDIDIA Pascal	2020
Orion Camera Controller Artemis I (EM-1)	Linux	Intel i5 CPU (NUC)	2021
Lunar IceCube	Linux	P400	2021
MX-1, MX-2, MX-5, MX-9 (CLPS) Moon Express	Unk	Unk	2021
Peregrine Lander - Commercial Lunar Payload Services (CLPS) Astrobotic Technology	Unk	Unk	2021
BioSentinel	VxWorks	UT700 LEON 3FT	2021
Orion Vision Processing Unit Artemis I (EM-1)	VxWorks	UT700 LEON 3FT	2021
SkyFire	Unk	Unk	2021
XL-1 Lander (CLPS) Masten Space Systems, Inc.	Unk	Unk	2021
Orion Camera Controller Artemis II (EM-2)	Linux	Intel i5 CPU (NUC)	2022
Plankton, Aerosol, Cloud, ocean, Ecosystem (PACE)	VxWorks	MUSTANG (custom LEON3 Dual core + LEON3-FT i	2022
Orion Vision Processing Unit Artemis II (EM-2)	VxWorks	UT700 LEON 3FT	2022
Lunar Gateway - Minimal Habitation Module (formerly Utilization Module)	VxWorks	Unk	2023
Exploration Extra-vehicular Mobility Unit (xEMU) Caution & Warning System (CWS)	VxWorks	Leon3 SPARC processor	2023
Roman Space Telescope (RST) (previously Wide-Field Infrared Survey Telescope - WFIRST)	RTEMS	Custom LEON4	2025
Mars Ascent Vehicle (preliminary)	Not App.	Sphinx	
Ames Modular Common Spacecraft Bus	VxWorks	RAD750 processor, 1GB TMR NVRAM	
Avionics & Software Platform for Exploration Capabilities & Technologies (ASPECT)	VxWorks	SP0 processor PowerQUICC III processor	
Certification of cFE on VxWorks ARINC-653	VxWorks ARINC 653	SP0 processor PowerQUICC III processor	

Table 7 – Projects and Programs Applying cFS Code

- There are indicia of unresolved problem reports being managed by the cFS project, but not considered by cFS's potential future users. Agency guiding documents include the expectation for OTS software users to periodically review OTS FSW defect backlogs for defect impacts[25]. Although not a subject addressed with surveyed projects, an observed project was both unaware of the expectation to review the backlog and had not internally developed a similar practice.
    - The gap created between requirements for reviewing the developers' OTS SW defect backlogs and project's execution highlighted another aspect of OTS SW reuse incompletely considered by the user community. The requirement to review the OTS developer's defect backlog was inserted into NASA's Software Engineering Requirements in Revision B, and made explicitly applicable to Open Source Software in Revision C.
- Table 8 – Project/Program Launch Dates within Epochs of NPR-7150.2 "NASA Software Engineering Requirements" provides a graphical representation of the launch history of cFS based projects and the period when each revision of the NASA Software Engineering Requirements document was applicable. Each NPR 7150.2 revision represents a modification of leadership's instructions for SW engineering, potentially including revisions to V&V expectations. This rate of change may impact (i) the ability for future missions to reuse cFS SW and (ii) for those same future projects to apply legacy V&V evidence to software assurance.
- Figure 8 demonstrates the gap between legacy SW engineering processes and evolving NASA requirements.[26] Green blocks reflect applicable requirements for a particular document revision.
    - Vertical columns with white bottoms and green above that represent requirements deleted.
    - Columns green at the base and white above reflect added requirements.
    - While the total number of requirements have remained relatively consistent across all four releases, the requirements have changed substantially.

Results were briefed to the NASA community via a Webinar sponsored by the NASA Safety Center (NSC) and held on June 17, 2020. The presentation slides and a video of the Webinar are available at: https://nsc.nasa.gov/events/detail/cfs-verification-and-validation-within-nasa-projects-and-programs?utm_source=cFS+Verification+and+Validation+Within+NASA+Projects+and+Programs&utm_medium=newsletter&utm_campaign=cFS+Verification+and+Validation+WIthin+NASA+Projects+and+Programs+Survey.

[25] Reference NPR 7150.2 Rev. B & Rev. C SWE-027 sub f "The *project manager shall satisfy the following conditions when a COTS, GOTS, MOTS, or reused software component is acquired or used: … f. The project has a plan to perform periodic assessments of vendor reported defects to ensure the defects do not impact the selected software components.*

[26] It is not intended to suggest that programs within a given epoch would have conformed to all the NPR 7150.2 requirements of that release. Project formulation and execution represents a substantial delay between the release of a requirement and the conformance of the community to new requirements.

	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
Avionics & Software Platform for Exploration Capabilities & Technologies (ASPECT)																						
Ames Modular Common Spacecraft Bus																						
Mars Ascent Vehicle (preliminary)																						
CATALYST (3 partners) 2014			NPR 7150.2 IR					NPR 7150.2 A					NPR 7150.2 B					NPR 7150.2 C			
Certification of cFE on VxWorks ARINC-653																						
Roman Space Telescope (formerly Wide-Field Infrared Survey Telescope - WFIRST)																						
Lunar Gateway - Minimal Habitation Module (formerly Utilization Module)																					X	
Exploration Extra-vehicular Mobility Unit (xEMU) Caution & Warning System (CWS)																					X	
Orion Vision Processing Unit Artemis II (EM-2)																				X		
Orion Camera Controller Artemis II (EM-2)																				X		
Plankton, Aerosol, Cloud, ocean, Ecosystem (PACE)																				X		
Lunar Gateway																				X		
Peregrine Lander - Commercial Lunar Payload Services (CLPS) Astrobotic Technology																			X			
XL-1 Lander (CLPS) Masten Space Systems, Inc.																			X			
MX-1, MX-2, MX-5, MX-9 (CLPS) Moon Express																			X			
BioSentinel																			X			
Orion Vision Processing Unit Artemis I (EM-1)																			X			
Orion Camera Controller Artemis I (EM-1)																			X			
Lunar IceCube																			X			
SkyFire																			X			
Int-Ball2																		X				
Orion Ascent Abort 2 (AA-2)																	X					
Coordinated Applied Capitol Technology University Satellite (CACTUS-1)																	X					
Space Test Program -Houston 6, USAF-NASA Goddard																	X					
Seeker ISS Flight Experiment																	X					
Kenobi ISS Flight Experiment																	X					
Simulation-to-Flight 1 (STF-1)																X						
Compact Radiation Belt Explorer																X						
Global Ecosystem Dynamics Investigation (at ISS) (GEDI)																X						
Parker Solar Probe (PSP)																X						
Dellingr															X							
Neutron star Interior Composition Explorer (NICER)															X							
Magnetospheric Multiscale Mission (MMS)													X									
Orion Exploration Flight Test 1 (EFT-1)												X										
Observatory for Planetary Investigations from the Stratosphere (OPIS)												X										
Global Precipitation Measurement (GPM) mission												X										
Lunar Atmosphere and Dust Environment Explorer (LADEE)											X											
Radiation Belt Storm Probes (RBSP) / Van Allen Probes										X												
Morpheus									X													
Lunar Reconnaissance Orbiter (LRO)							X															
CHIPSat (Precursor)	X																					

Table 8 – Project/Program Launch Dates within Epochs of NPR-7150.2 "NASA Software Engineering Requirements"

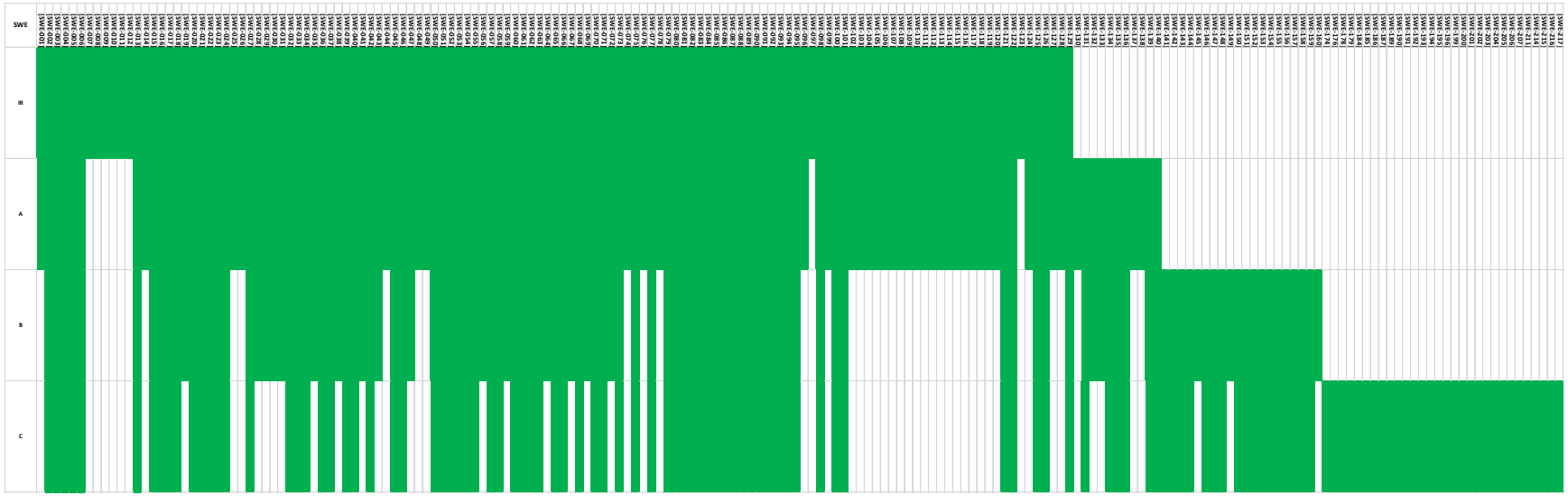


Figure 8 – NPR 7150.2 SWE Additions and Deletions (Horizontal) Per Document Revision (Vertical)

Project	SW Class	NASA Center	$	OS	Platform	Launch Date	Note
CHIPSat (Precursor)	D					1/12/2003	
Lunar Reconnaissance Orbiter (LRO)	B	GSFC	XXXM	VXWorks	RAD750	6/18/2009	Completed primary mission. Continues.
Morpheus	C (SC)	JSC	XXM	VxWorks	AiTech S950 (PPC750FX)	4/25/2011	https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20140001490.pdf
Radiation Belt Storm Probes (RBSP) / Van Allen Probes	B			VxWorks	RAD750	8/30/2012	End of mission 10/18/2019
Lunar Atmosphere and Dust Environment Explorer (LADEE)	B	ARC		VxWorks		9/7/2013	End of mission 4/17/2014
Global Precipitation Measurement (GPM) mission	B	GSFC	XXXM	VXWorks	RAD750	2/27/2014	Elapsed mission 6 yrs, 2 months of 3 yr plan.
Observatory for Planetary Investigations from the Stratosphere (OPIS)	D	GSFC		Xenomai	Intel Duo	10/8/2014	End of mission 10/08/2014
Magnetospheric Multiscale Mission (MMS)	B	GSFC	XXXM	RTEMS	Rad Hard Coldfire (5208)	3/12/2015	Elapsed mission 5 yrs, 1 month of 2.5 yr plan.
Neutron star Interior Composition Explorer (NICER)	D	GSFC		VxWorks	BRE440 PowerPC	6/12/2017	ISS weekly science posted to web.
Dellingr	D	GSFC		FreeRTOS	Gomspace Nanomind A712d ARM processor	8/14/2017	https://www.nasa.gov/feature/goddard/2018/dellingr-the-little-cubesat-that-could
Parker Solar Probe (PSP)	B	APL		RTEMS	LEON3 UT699	8/6/2018	April 29, 2020 mission update.
Global Ecosystem Dynamics Investigation (GEDI)	D			VxWorks	PowerPC 440	12/5/2018	
Simulation-to-Flight 1 (STF-1)	D	GSFC		FreeRTOS 9	Gomspace Nanomind A3200	12/16/2018	… objective was achieved …zero FSW errors… 600 days of STF-1 operations… Mission continues
Compact Radiation Belt Explorer (CeREs)	B	GSFC		Linux	CHREC Space Processor Instrument Computer Xlinux Zynq – ARM A9	12/16/2018	
Seeker ISS Flight Experiment	C (nSC)	JSC		Wumbo GNU/ Linux	CHREC space processor	04/17/2019	https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20190000520.pdf
Kenobi ISS Flight Experiment	Unk					4/17/2019	
Space Test Program Houston 6 (STP-H6)	Unk	GSFC				5/4/2019	
Coordinated Applied Capitol Technology University Satellite (CACTUS-1)	D		CubeSat			6/1/2019	
Orion Ascent Abort 2 (AA-2)	B	JSC	XXXM	VxWorks	AiTech SP0 1 GHz SBC	7/2/2019	Completed. No cFS anomalies.
BioSentinel	D	ARC		VxWorks	UT700	NET 4/18/2021 (EM-1)	
Orion Vision Processing Unit Artemis I (EM-1)	A	JSC	XXB	VxWorks	UT700 LEON3FT	NET 4/18/2021	

Orion Camera Controller Artemis I (EM-1)	**A**	**JSC**		**Ubuntu-64 Linux**	**Intel i5 CPU (NUC)**	**NET 4/18/2021**	
Lunar IceCube	**B**	**GSFC**		**Linux**	**P400**	**NET 4/18/2021**	
SkyFire	**D**					**NET 4/18/2021**	
Peregrine Lander - Commercial Lunar Payload Services (CLPS) Astrobotic Technology	**Unk**	**NASA HQ**				**NET 2021**	
XL-1 Lander (CLPS) Masten Space Systems, Inc.	**Unk**	**NASA HQ**		**Linux (implied)**		**NET 2021**	
MX-1, MX-2, MX-5, MX-9 (CLPS) Moon Express	**Unk**	**NASA HQ**				**NET 2021**	
Lunar Gateway (Multiple Projects)	**A**		**XXB**			**NET 2022**	
Plankton, Aerosol, Cloud, ocean, Ecosystem (PACE)	**B**	**GSFC**	**XXXM**	**VxWorks**	**Custom LEON3 Dual core + LEON3FT in RTG4 FPGA**	**8/18/2022**	
Orion Camera Vision Processing Unit Artemis II (EM-2)	**A**	**JSC**	**XXB**	**VxWorks**	**UT700 LEON (LEON3FT)**	**10/1/2022**	
Orion Camera Controller Artemis II (EM-2)	**A**	**JSC**	**XXB**	**Ubuntu-64 Linux**	**Intel i5 CPU (NUC)**	**10/1/2022**	
Advanced Exploration Extra-vehicular Mobility Unit (Exploration EMU) Caution & Warning System (CWS)	**A (SC)**	**JSC**		**VxWorks**	**LEON3 SPARC processor**	**NET 2023**	
Lunar Gateway - Minimal Habitation Module (formerly Utilization Module)	**A**	**JSC**	**XXXM**	**VxWorks**		**NET 2024**	
Roman Space Telescope (RST) (formerly Wide-Field Infrared Survey Telescope - WFIRST)	**B**	**GSFC**	**XB**	**RTEMS**	**Custom LEON4**	**mid-2020s**	
CATALYST (3 partners) 2014	**NA**		**NA**			**Not Applicable**	
Certification of cFE on VxWorks ARINC-653	**A**	**JSC**		**VxWorks ARINC-653**	**SP0 processor**	**Not Applicable**	
Mars Ascent Vehicle (preliminary)	**NA**	**MSFC**		**RTEMS**	**Sphinx**	**2026 or 2031**	
Int-Ball2	**NA**	**JAXA**		**Linux**	**JETSON TX2 NVDIDIA Pascal**	**NET 2020**	
Ames Modular Common Spacecraft Bus	**Unk**	**ARC**		**VxWorks OS**	**RAD750 processor, 1GB TMR NVRAM**	**Not Applicable**	
Avionics & Software Platform for Exploration Capabilities & Technologies (ASPECT)	C (nSC)	JSC	XXM	VxWorks	SP0 processor	2019	

Table 9 – Identified Projects and Programs Applying cFS Code

4 Conclusions and Recommendations

The analyses completed were positive and have established a number of observations:

- The cFS user base is large when compared to the previous approaches applying either unique developments or limited development "Clone and Own" approaches.
    - The user base and mission rate is too small to establish highly mature and stable code that fully addresses management expectations.
    - The pace of changes to NPR 7150.2 NASA Software Engineering Requirements will serve as one limit on the reuse of verification and validation evidence as leadership expectations evolve.
- There are a significant number of projects successfully applying the cFS code as shown in Table 9 above.
    - A significant number of projects have reached an operational stage, without a failure publicly attributed to cFS.
        - A number of projects have completed their scheduled mission life.
    - Adoption of cFS continues, and the rate of adoption appears to be increasing.
    - cFS use spans a wide range of software classifications and NASA centers.
    - Differences in Operating Systems and processing hardware serve as evidence that the cFS approach to incorporating abstraction layers is sound and cFS OTS products are generally well integrated into NASA programs.
- "Families" of similar OS SW and HW instances limit the total novelty between instances and limit architecture risk.
    - There does not appear to be a move to consolidate HW/SW configurations to a limited set.
    - V&V reuse is unlikely to be the direct reuse of "approved configurations" with reusable V&V evidence packages or a "certified" configuration.
- Reuse of cFS verification and validation evidence appears reasonable and capable of (i) meeting leadership V&V expectations, (ii) informing the cFS user community, (iii) increasing SW maturity through increased value added testing and (iv) eliminating unnecessary activity. SARP Open Source cFS FSW V&V project observations on the reusability of cFS V&V processes, tools, and results are provided in Appendix B..
- Shared characteristics of flight software and multiple projects incorporating cFS have not resulted in transferable software assurance evidence while surveyed projects have limited cFS V&V in response to perceptions of lowered risk and increased code maturity.
    - Evidence used within cFS assurance could be transferable, but this approach has not been observed within the NASA cFS software community.
    - Building libraries of reusable cFS evidence for specific aspects of V&V is possible and would result in improved cost certainty and cost control. The following representative V&V activities could be made portable across cFS projects: (i) static code analysis results, (ii) verification of cFS and cFS application requirement implementation, (iii) verification of algorithms, (iv) requirements' quality characteristics and completeness, and (v) requirement traces to design, implementation and test, etc.

        - The cFS applications analyzed for this activity demonstrated that the documentation of the cFS design was generally insufficient both to assure the design and support future code maintenance.
        - The cFS static code analysis found that with the exception of string processing, the cFS software is generally free of the kinds of potential bugs that SCA readily identifies. The source code generally holds to standards that lead to understandable and maintainable software while string processing related warnings suggest individual targets for code improvements.
- Projects surveyed had confidence in the cFS code and perceived it to be both stable and mature.
    - Multiple Class A or B projects are not verifying their instance of cFS.
    - Multiple SW projects were not methodically examining available evidence for completed V&V activities where assurance evidence can be reused or risk could be identified.
    - Observed projects were performing validation of the cFS through incorporation of the cFS within the larger system during mission-specific application verification testing and verification testing at the subsystem or high level of assembly.
    - Class "B" cFS reuse and expansion of the user community to Class "A" safety-critical SW is at least partially based on the perception of cFS success within Class "B" uses rather than detailed consideration of cFS assurance evidence.
    - Some projects appear to be attempting nearly comprehensive verification and validation of their cFS instance.
- Unresolved problem reports are being managed by the cFS project.
- Unresolved problem reports are not considered by cFS's potential future users. NASA Software Engineering Requirements makes such reviews mandatory for Class A, B, C, and D software projects.

The SARP Open Source cFS FSW V&V project identified a candidate risk to NASA flight assurance goals from external user's perception of cFS maturity, stability, and low risk driving cFS acceptance without performing targeted verification and validation. This candidate risk, included in Appendix C., has been provided to the NASA Technical Fellow for Software Assurance, Office of Safety and Mission Assurance (OSMA) for consideration as an agency-level risk.

The SARP project also identified four potential candidate risks associated with projects' use of cFS middleware that will have various scores for risk consequence and risk likelihood based on project-specific evaluation. Guidance for establishing the cFS middleware risk consequence and likelihood is provided in Appendix D.. These candidate risks, unlike the agency-level candidate risk mentioned above, potentially apply to each project using cFS middleware and may have different risk consequence and risk likelihood scores from project to project.

Appendix A. Acronyms

Acronym	Description
AA-2	Ascent Abort 2
AES	Advanced Exploration Systems
APL	Applied Physics Laboratory
ARC	Ames Research Center
ARINC	Aeronautical Radio, Incorporated
BFS	Backup Flight Software
CC	Camera Controller
CCSDS	Consultative Committee for Space Data Systems
CF	CFDP File application
CFDP	CCSDS File Delivery Protocol
CFE	Core Flight Executive
cFS	core Flight System
COTS	Commercial Off The Shelf
CS	Coding Standard
CS	Checksum application
CSCI	Computer Software Configuration Item
CWE	Common Weakness Enumeration
DS	Data Store application
EAR	Export Administration Regulations
ECR	Engineering Change Request
EM-1	Exploration Mission 1
FIPS	Federal Information Processing Standards
FM	File Manager application
FSW	Flight Software
FP	False Positive
FY	Fiscal Year
GOTS	Government Off The Shelf
GSFC	Goddard Space Flight Center
GPM	Global Precipitation Measurement
HK	Housekeeping application
HS	Health and Safety application
HW	Hardware
IDD	Interface Design Document
IO	Input/Output
IR	Initial Release
IRS	Interface Requirements Specification
ITAR	International Traffic in Arms Regulations
IV&V	Independent Verification and Validation
JAXA	Japan Aerospace Exploration Agency
JSC	Johnson Space Center
LC	Limit Checker application

Acronym	Description
LRO	Lunar Reconnaissance Orbiter
MD	Memory Dwell application
MM	Memory Manager application
MOTS	Modified Off The Shelf
MPCV	Multi-Purpose Crew Vehicle
MSFC	Marshall Space Flight Center
NA	Not Applicable
NASA	National Aeronautics and Space Administration
NPR	NASA Procedural Requirements
NSC	NASA Safety Center
OS	Operating System
OSAL	OS Abstraction Layer
OSMA	Office of Safety and Mission Assurance
OSS	Open Source Software
OTS	Off-The-Shelf
PB	Potential Bug
PROM	Programmable Read Only Memory
RBSP	Radiation Belt Storm Probe
RTEMS	Real-Time Executive for Multiprocessor Systems
RTOS	Real Time Operating System
SA	Software Assurance
SARP	Software Assurance Research Program
SBN	Software Bus Network application
SC	Stored Command application
SCA	Static Code Analysis
SCH	Scheduler application
SDD	Software Design Document
SEI	Software Engineering Institute
SRS	Software Requirements Specification
SW	Software
SWE	Software Engineering Requirement
TA	Technical Authority
V&V	Verification and Validation
VPU	Vision Processing Unit

Appendix B. cFS Verification and Validation Reuse

The reusability of cFS V&V processes, tools, and results considered only the cFS related aspect of this V&V effort. cFS SW incorporated into a larger system would require consideration of all code. In a limited number of cases (e.g., static code analysis of cFS source code), the process, tools, and results of prior analyses can be reused. Establishing a pattern of reuse will inform the user community more effectively than observed users who have defined cFS out of their V&V approach based on the perception of success and code stability.

All identified V&V processes below from the NASA IV&V Technical Framework[27] are reusable at a level similar to NASA IV&V process methods documented within the "COMPASS" tool (https://compass.ivv.nasa.gov/). Method tailoring to a cFS-specific tool is possible. Additional tool reuse (primarily test environments, test scripts, or test cases) has already been observed within the cFS community. Enhancements to these approaches as well as expansion to the reuse of assurance evidence (test results, static code analysis results, etc.) appears to be reasonable next steps.

Evaluations reported below are limited to the cFS software. Program/project developed code, integrated, or configured software was not addressed.

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
2.0	Verify and Validate Concept Documentation. Concept documentation represents the delineation of a specific implementation solution to solve the acquirer's problem. The objective of Concept IV&V is to validate the selected solution and ensure that no false assumptions have been incorporated in the solution. Additional objectives:	Partial - Tools for evaluating concept are reusable at high level (similar to NASA IV&V process asset library). Partial reuse of tailored cFS specific tools is possible. Details are addressed as part of Section 2.1 through 2.9 below.	None
2.1	Ensure that software planned for reuse meets the fit, form, and function, and security as a component within the new application.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	Partial.
2.2	Ensure that the system architecture contains the necessary computing related items (subsystems, components, etc.) to carry out the mission of the system and satisfy user needs and operational scenarios or use cases.	Partial (See 2.0 above for limitations)	Partial. Families of HW/SW can suggest risk, list prior successes, and aid in defining cFS based system capability within a given architecture.
2.3	Ensure that the concepts for the operations, mission objectives (including mission retirement), and the system are sufficiently defined as a basis for the engineering and planning of computing related functions.	Partial (See 2.0 above for limitations)	None
2.4	Ensure that feasibility studies provide the results necessary to confidently support the key decisions that drove the need for the study.	Partial (See 2.0 above for limitations)	Partial. Families of HW/SW can suggest risk and list prior successes. Closely related HW/SW and mission architecture may represent alternatives to aspects of feasibility studies.
2.5	Ensure that known software based hazard causes, contributors, and controls are identified and documented.	Partial (See 2.0 above for limitations)	Yes - for hazards related to cFS.

[27] https://www.nasa.gov/sites/default/files/atoms/files/ivv_09-1_independent_verification_and_validation_technical_framework_-_ver_p_-_10-25-2017.pdf

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
2.6	Ensure that security threats and risks are known, up to date, appropriately documented, and are correct for this mission and that relevant regulatory requirements are identified.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None - the communication of cFS open source software related SW vulnerabilities was not addressed. The need to (i) protect cFS vulnerabilities from disclosure to hostile actors and (ii) communicate vulnerabilities to NASA users might be a permanent system weakness. The absence of a comprehensively documented user community appear to be part of a future secured communications path.
2.7	Ensure that appropriate plans are in place to update the security threats and risks over the course of the development lifecycle to allow for introduction of new or changing threats, and are consistent with project data categorization (e.g. FIPS).	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None.
2.8	Ensure the security risks introduced by the system itself, as well as those associated with the environment with which the system interfaces, are appropriately accounted for in the known threats.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None
2.9	Ensure the system concept from a security perspective and assure that potential security risks with respect to confidentiality (disclosure of sensitive information/data), integrity (modification of information/data), availability (withholding of information or services), and accountability (attributing actions to an individual/ process) have been identified. Include an assessment of the sensitivity of the information/data to be processed and assessment of its consistency with FIPS categorization.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None
3.0	Verify and Validate Requirements. Requirements IV&V addresses a system's software requirements including analysis of the functional and performance requirements, interfaces external to the software, and requirements for qualification, safety and security, dependability, human factors engineering, data definitions, user documentation for the software, installation and acceptance, user operation and execution, and user maintenance. The objective of Requirements IV&V is to ensure the system's software requirements are high quality (correct, consistent, complete, accurate, unambiguous, and verifiable), and will adequately meet the needs of the system and expectations of its customers and users, considering its operational environment under nominal and off-nominal conditions, and that no unintended features are introduced (see Key Concepts 1 and 2, above). Additional objectives:	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project. Details are addressed as part of Section 3.1 through 3.7 below.	None
3.1	Ensure that the system requirements are of high quality and are consistent with acquirer needs as they relate to the system's software.	Yes (limited to cFS aspects)	No. Application of this requirement to only aspects of system requirements necessary for hosting cFS is possible. This definition of scope was rejected for this report.
3.2	Ensure that all (in-scope) parent requirements are represented in the appropriate child requirements and that the child requirements do not introduce capability that is not required.	Partial (See 2.0 above for limitations)	There will be unnecessary requirements in many uses of OTS SW. These unnecessary requirements reflect capabilities implemented in code but not essential to operations.
3.3	Ensure that the software requirements are of high quality and adequately meet the needs of the system with respect to expectations of its customer and users, operational environment, and both functional and non-functional perspectives.	Partial (See 2.0 above for limitations)	Partial. Analysis of requirement quality characteristics is largely insensitive to planned use and draws heavily on OTS SW's OTS requirements.
3.4	Ensure that the requirements for software interfaces with hardware, user, operator,	Partial (See 2.0 above for limitations)	Partial. OTS SW's OTS requirements only

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
	and other systems are adequate to meet the needs of the system with respect to expectations of its customer and users, operational environment, dependability and fault tolerance, and both functional and non-functional perspectives.		partially document the intended interfaces.
3.5	Ensure that software requirements meet the dependability and fault tolerance required by the system and provide the capability of controlling identified hazards and do not create hazardous conditions.	Partial (See 2.0 above for limitations)	None. Dependability, fault tolerance, and hazard control are implementation specific. Even if addressed by a common executable code, will be driven by configurable data and selected architecture.
3.6	Ensure that the requirements address the security threats and risks identified within the system concept specifications and/or the system security concept of operations (e.g. System Security Plan).	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None. Program/project specifics will drive requirements. OTS requirements analyzed did not incorporate an associated baseline.
3.7	Ensure that requirements define appropriate security controls to the system, subsystem, according to NPR 2810 and driven by the Project's security needs and requirements.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None. Program/project specifics will drive requirements. OTS requirements analyzed did not incorporate an associated baseline.
4.0	Verify and Validate Test Documentation. Test Content IV&V addresses test plans, procedures, cases, and designs. The objective is to ensure that the collection of test related content will serve as a sufficient means to verify and validate that the implementation meets the requirements and operational need under nominal and off-nominal conditions (see Key Concepts 1 and 2 above). Test content should be evaluated for requirements coverage and test completeness, considering the extent of the software exercised, the appropriateness of the verification method (e.g. test, analysis, demonstration, inspection), whether the set of inputs used during testing are a fair representative sample from the set of all possible inputs to the software, and whether test inputs include boundary condition inputs, rarely encountered inputs, invalid inputs, inputs related to identified hazards, safety and security of the software and system. Additional objectives:	Partial (See 2.0 above for limitations). Reusable test cases and scripts are already shared by the cFS user community. Details are addressed as part of Section 4.1 through 4.9 below.	
4.1	Ensure that the planned tests are sufficient to:	NA	NA
4.1.1	Ensure that the software correctly implements system, software, and security requirements in an operational environment under nominal and off-nominal conditions.	Partial (See 2.0 above for limitations).	Partial.
4.1.2	Ensure that the complete, integrated system complies with its specified system requirements allocated to software and to validate whether the system meets its original objectives.	Partial (See 2.0 above for limitations).	None
4.1.3	Ensure that the software meets all of the (in-scope) software requirements and is ready to be integrated with system hardware.	Partial (See 2.0 above for limitations). Reusable test cases and scripts are already shared by the cFS user community.	None
4.1.4	Ensure that the software correctly and securely implements the software requirements and design as each software component (e.g., units or modules) is incrementally integrated with each other.	Partial (See 2.0 above for limitations). Pre-delivery testing by cFS development teams has been demonstrated.	Partial. Pre-delivery testing by cFS development teams has been demonstrated.
4.1.5	Ensure that the software components (e.g., units, source code modules) correctly implement software component requirements.	Partial (See 2.0 above for limitations). Reusable test cases and scripts are already shared by the cFS user community.	Partial. Primarily testing at the unit level or below. Variations in system configuration (operating system, hardware, configured software, etc.) currently limits direct evidence reuse above the unit level or of component

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
			interface requirements.
4.2	Ensure that valid relationships are defined between the Test Plans, Designs, Cases, and Procedures for test types and documents subject to IV&V test analysis.	Partial (See 2.0 above for limitations). Reusable test cases and scripts are already shared by the cFS user community.	Partial. Primarily testing at the unit level or below. Variations in system configuration (operating system, hardware, configured software, etc.) currently limits direct evidence reuse above the unit level or of component interface requirements.
4.3	Ensure that the planned regression testing to be performed when changes are made to any previously examined software products is sufficient to identify any unintended side effects or impacts of the change on other aspects of the system (including not increasing the security risk).	Partial (See 2.0 above for limitations)	Partial. Pre-delivery retesting of CSCI level requirements after ECR updates have been noted. Regression testing at higher levels of assembly would occur within cFS users project/program testing.
4.4	Ensure that any simulations are sufficiently complete, correct, and accurate to perform the intended testing.	Partial (See 2.0 above for limitations). Simulation reuse or sharing has been observed.	Partial. Simulation sharing has been observed.
4.5	Ensure that the Test Cases under analysis:	NA	NA
4.5.1	Specify the correct test inputs, predicted results, and sets of execution conditions necessary to satisfy their intended test objectives (covering both nominal and off-nominal conditions)	Partial (See 2.0 above for limitations)	Partial. cFS tests to be performed at the CSCI level or below could be structured to be reusable. cFS tests from higher levels of test, configured systems and interface requirements would not be reusable.
4.5.2	Verify specific security controls (physical, procedural and automated controls) cannot be breached leading to compromise of information confidentiality, integrity, or availability.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None
4.6	Ensure that the Test Procedures under analysis specify the correct sequence of actions necessary for the execution of the tests to satisfy their intended test objectives.	Partial (See 2.0 above for limitations)	Partial. cFS tests to be performed at the CSCI level or below could be structured to be reusable. cFS tests from higher levels of test, configured systems and interface requirements would not be reusable.
4.7	Ensure that the Test Designs under analysis correctly specify the details of the test approach for the covered software feature or combination of software features and identify the associated tests.	Partial (See 2.0 above for limitations)	Partial. cFS tests to be performed at the CSCI level or below could be structured to be reusable. cFS tests from higher levels of test, configured systems and interface requirements would not be reusable.
4.8	Ensure that the test environment is sufficiently complete, correct, and accurate to perform the intended testing.	Partial (See 2.0 above for limitations).	Partial. cFS tests to be performed at the CSCI level or below could be structured to be reusable. cFS tests from higher levels of test, configured systems and interface requirements would not be reusable or have limited portability.
4.9	Ensure that the integrated system testing covers any areas that may potentially increase the security risk,	Cyber security was not addressed as part of this project.	None. Cyber security was not addressed as part of this project. Interface testing (necessary for the V&V activity) requires greater levels of implementation specific details and therefore will display greater levels of variability, reducing

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
			reuse.
5.0	Verify and Validate Design. In software design, software requirements are transformed into an architecture and a detailed design for each software component. The design also includes databases and system interfaces (e.g., hardware, operator/user, software components, and subsystems). Design IV&V addresses software architectural design and software detailed design. The objective of Design IV&V is to ensure that the design is a correct, accurate, and complete transformation of the software requirements that will meet the operational need under nominal and off-nominal conditions, that no unintended features are introduced, and that design choices do not result in unacceptable operational risk (see Key Concepts 1 and 2 above). Additional objectives:	Partial (See 2.0 above for limitations). Details are addressed as part of Section 5.1 through 5.8 below. cFS software design stability (reuse across multiple projects/programs) is a primary element of system success to date. However, user demands will vary and will impact design V&V and V&V tool complexity.	Details are addressed as part of Section 5.1 through 5.8 below. cFS design documentation is generally very modest. cFS architecture at the system and CSCI level was generally good. cFS design documentation at the CSCI level and down was generally very limited. User project/program V&V activities in this area may not be equal to similar developed systems. The fact that cFS is OTS has limited the impact of this weakness on users. Consideration of future design V&V would need to include assessments in this area.
5.1	Ensure that all (in-scope) requirements (e.g. SRS and IRS) are represented in the appropriate elements of the design (e.g. SDD and IDD) and that the design does not introduce capability that is not required.	Partial. The design V&V tool reuse would be very high for cFS design at or below the CSCI level and excluding interfaces and configurable elements of design.	For cFS design at or below the CSCI level and excluding interfaces and configurable elements of design, design assurance reuse would be high.
5.2	Ensure that the design provides the required capability (meeting software architecture, software security, and software requirements), is able to reliably meet user needs, and is sufficiently stable to proceed with implementation.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project. Sufficient variations in architecture and user needs may drive tailored or unique V&V activities/tools.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project. Few explicit security requirements were noted in the cFS SW requirements . For cFS design at or below the CSCI level and excluding interfaces and configurable elements of design, design assurance reuse would be very high. Sufficient variations in architecture and user needs will drive associated V&V activities.
5.3	Ensure that the proposed software architecture satisfies the needs of the system, and that it is a feasible solution (i.e. will successfully satisfy the needs of the system, while still being practical).	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project. Variations in architecture and user needs may drive tailored or unique V&V activities/tools.	Partial. Families of successful HW/SW configurations suggest limited risk when applied to new projects. Residual risk may be limited to areas of processing capacities, interfaces and security (depending on mission risk and classification).
5.4	Ensure that the internal and external software interface designs are provided for all (in-scope) interfaces with hardware, user, operator, software, and other systems and that they provide sufficient detail to enable the development of software components that implement the interfaces.	Partial. Abstractions (such as the software bus) ease the development of project unique interfaces such as IO. V&V tool requirements may be simple and consequently very reusable.	Partial (See 2.0 above for limitations). cFS abstractions (such as the software bus) ease the development of project unique interfaces such as IO. Risk in this area might be very low for some projects.
5.5	Ensure that complex algorithms have been correctly derived, provide the needed behavior under off nominal conditions and assumed conditions, and that the derivation approach is known and understood to support future maintenance.	Abstractions (such as the software bus) ease the development of project unique interfaces such as IO. V&V tool requirements may be simple and consequently very reusable.	See comment in 5.0 above. If suitable design documentation was available and if the analyses were performed, algorithm design V&V reuse at the CSCI level and down would be very high.
5.6	Ensure that the design provides the dependability and fault tolerance required by the system and that the design is capable of controlling identified hazards and does not create hazardous conditions.	Partial (See 2.0 above for limitations)	Partial.
5.7	Ensure that the architecture and detailed design adequately address the identified	Partial (See 2.0 above for limitations). Cyber	Cyber security was not addressed as part of this

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
	security requirements both for the system and security risks, including the integration with external components and information and data utilized, stored, and transmitted through the system.	security was not addressed as part of this project.	project.
5.8	Ensure that identified security threats and vulnerabilities are prevented, controlled, or mitigated via proposed design components. Any unmitigated threats and vulnerabilities are documented and addressed as part of the system and software operations.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	Cyber security was not addressed as part of this project.
6.0	Verify and Validate Implementation. In software implementation, the design is transformed into code, database structures, and related machine executable representations. The objective of Implementation IV&V is to verify and validate that these transformations are correct, accurate, and complete, yielding source code that correctly implements requirements, meets the operational need under nominal and off-nominal conditions, and introduces no unintended features (see Key Concepts 1 and 2 above). Implementation IV&V also seeks to ensure that the source code and documentation (both embedded and stand-alone) are complete and provide an adequate reference for source code maintainability and upgrade. Additional objectives:	Partial (See 2.0 above for limitations). Details are addressed as part of Section 6.1 through 6.8 below. . cFS software code stability (reuse across multiple projects/programs via configurable parameters) is a primary element of system success to date.	cFS code quality (readability, structure, maturity, etc.) was noted to be very high. This would ease code V&V activities. However, user demands will vary and will impact implementation V&V.
6.1	Ensure that all (in-scope) elements of the design (e.g. SDD and IDD) are represented in the appropriate source code components and that the source code does not introduce capability that is not required.	Partial. See comment in 5.0 discussing the limitation of the software design documentation details experienced in this project. Limited design documentation substantially impacts the ability to perform this V&V activity.	See comment in 5.0 discussing the limitation of the software design documentation details experienced in this project.
6.1.1	Ensure that the implementation adheres to the system and software design in that it addresses the identified security risks and that the implementation does not introduce new security risks through specific code constructs, features, or coding flaws (e.g. Common Weakness Enumerations).	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project. Mapping of CWE to static code detectable errors has been previously demonstrated. Reuse of both static code analysis approaches and static code analysis results of unchanged code has previously been demonstrated.	Partial (See above for limitations). Security was not considered as part of this project. Mapping of CWE to static code detectable errors has been previously demonstrated. Reuse of both static code analysis approaches and static code analysis results of unchanged code has previously been demonstrated.
6.2	Ensure that the source code components can reliably perform required capabilities under nominal and off-nominal conditions, perform no undesired behaviors, and that the documentation (both embedded and stand-alone) can facilitate code maintenance.	Reusable	Partially reusable. Results of Static Code Analysis (SCA) (e.g. IV&V method M-9 "Verify Software Code Quality using Static Analysis Tools") has a proven history of portability. IV&V projects historically avoid re-performing SCA performed previously by IV&V or the project/program supported because of the reusability of this assurance evidence.
6.3	Ensure that the source code that interfaces with hardware, user, operator, software, and other systems reliably provides the right services and data and receives data for internal use.	Partial. Interfaces implementation has been driven by project/program details and the analyses of such code is likely to also be driven by project/program details.	None
6.4	Ensure that test results are as expected (per the corresponding plans, cases, procedures, design) and the impacts of any discrepancies are understood.	Partial (See 2.0 above for limitations)	cFS tests to be performed at the CSCI level or below could be structured to be reusable. cFS tests from higher levels of test, configured systems and interface requirements would not be reusable.
6.5	Ensure that the source code components provide the dependability and fault tolerance	Partial (See 2.0 above for limitations)	None

Sect	Verification and Validation Activity	Tool Reuse	Evidence Reuse
	required by the system and that the source code is capable of controlling identified hazards and does not create hazardous conditions.		
6.6	Ensure that all (in-scope) requirements (e.g. SRS and IRS) are represented in the appropriate source code components and that the source code does not introduce capability that is not required.	Partial (See 2.0 above for limitations)	cFS tests to be performed at the CSCI level or below could be structured to be reusable. cFS tests from higher levels of test, configured systems and interface requirements would not be reusable.
6.7	Ensure that the system and software-required threat controls and safeguards are correctly implemented per proposed (or baselined) design components and validate that they provide the desired levels of protection against threats to the system. Any unmitigated threats and vulnerabilities are documented and addressed as part of the system and software operations.	Cyber security was not addressed as part of this project.	None
6.8	Ensure the appropriate level of data protection is defined and maintained across all instances and transactions throughout the system and that the security controls are defined to provide comprehensive (end-to-end) protection for the life of the data.	Partial (See 2.0 above for limitations). Cyber security was not addressed as part of this project.	None

Appendix C. Candidate Statement of cFS Verification and Validation Risk

The SARP Open Source cFS FSW V&V project identified a candidate risk to NASA flight assurance goals from external user's perception of cFS maturity, stability, and low risk driving cFS acceptance without performing targeted verification and validation. This candidate risk has been provided to the NASA Technical Fellow for Software Assurance, Office of Safety and Mission Assurance (OSMA) for consideration as an agency-level risk.

IV&V Risk ID	R-###	**Consequence**	5	**Likelihood**	3	**Priority**	21	**State**	Draft
Title		Lack of Project-Specific V&V Targeting cFS/CFE May Result in Significant Anomalies During Flight							
Risk Statement		[CONDITION] If projects using Core Flight System (cFS)/Core Flight Executive (CFE) assume falsely that cFS/CFE is stable and adequately verified and validated (V&V'ed) and therefore does not require project-specific V&V activities targeting cFS/CFE software, [DEPARTURE] there is a possibility that the version of cFS/CFE projects are using may contain errors which, when undetected through usage during normal V&V activities, [ASSET] affect the ability of cFS-equipped projects' software to respond appropriately under certain conditions, [CONSEQUENCE] thereby leading to the increased potential of anomalies during flight for projects using cFS, including possible loss of subsystem functionality, loss of mission or loss of life (potentially including loss of crew for Class A software systems).							
Context Statement		cFS is a NASA "open source" flight software framework and associated flight software (FSW) middleware, with levels of abstraction intended for multiple hardware/software environments. The user community expresses high product confidence, e.g. *"Fully tested, documented, operational with LRO spacecraft, several other operational missions since"*. This type of quote reflects an attitude of a single definition of what fully tested or fully certified means. It is a reflection of the origin of the cFS software more than a reflection of the current use of the software, and of a misperception that certification is generic for all possible uses. Certification applies to a specific instantiation of the software as used on a specific mission, and changes made to the software relative to the certified version or usage of the certified version under different mission scenarios invalidate that certification, requiring additional testing to certify the new version for the different mission. Similarly, once the software or its usage is changed, it is no longer "fully tested" until additional testing is performed. NASA use of cFS is increasing, including use on Class A Safety Critical missions. Multiple NASA Class A or B projects are not verifying their instance of cFS, but the identified record for intended cFS benefits did not include V&V reduction. Projects perceive cFS stability and successful prior use. NASA Procedural Requirements (NPR) 7150.2 "C" revision SWE-027 sub "e", "The (COTS, GOTS, MOTS, or reused) software component is verified and validated to the same level required to accept a similar developed software component for its intended use," is not applied. This comment from a Class C Project, *"Decreased verification*							

	needed - NO anomalies found in CFS code – TRL9", reflects the double-edged problem: less testing will increase the perception of the software being mature and not requiring tests. *"Conservatively saved 80+ man-years development time… …$20M".* Software Assurance Research Program (SARP) project "Open Source cFS FSW V&V" established that some FSW development teams were not periodically reviewing reported cFS defects which would be required by NASA NPR 7150.2C SWE-027 sub "f". This can result in known controllable defects being permitted to impact mission success or safety or uncontrolled defects not being properly understood by the user community of the projects using cFS. The perception of cFS success and maturity combined with an easy ability to re-execute prior generic cFS testing may retard the actual growth of cFS software maturity and increase the gap between perceived FSW risk and actual FSW risk. Programmatic risk acceptance requires informed consideration of actual risk.
Closure Criteria	Each development program using cFS/CFE either uses the latest V&V'ed version of cFS/CFE or assesses the potential impacts to their program of any defects identified in or changes made to cFS/CFE subsequent to the version they are using, AND performs project-specific V&V of changes to cFS/CFE that do affect their program. -OR- Assessment of risk being taken by the development program by treating cFS/CFE software as stable and fully verified and validated in order to meet cost and schedule demands of the program is performed and understood by OSMA.
Consequence	5
Consequence Rationale	Rationale for combined Consequence score of 5: (Performance and Safety) 5 - Within NASA programs, the failure to adequately verify software (or other products) allows for the possibility that the products will not work as required. FSW worst-case failures could potentially result in (a) the inability to achieve minimum success criteria, (b) loss of vehicle/mission, or (c) the loss of human life (on the ground or in flight) unless dissimilar SW or independent physical protection (mechanical controls, barriers, etc.) are present.
Likelihood	3
Likelihood Rationale	3 - Moderate. Controls exist with some limitations or uncertainties. Without project-specific V&V activities targeting cFS/CFE software, there is moderate likelihood of errors in the projects' version of cFS/CFE going undetected through usage during normal V&V activities and surfacing during flight.

Appendix D. Guidance For Establishing cFS Middleware Risk

Establishing the risk "likelihood" associated with any aspect of SW middleware is not different from establishing the likelihood of risks for any other OTS software. Establishing the risk "consequence" associated with any aspect of SW middleware is slightly more challenging, and similar challenges can be associated with establishing the risk consequence of using COTS Operating Systems.

- "Service" within this guidance is not limited to CFE Services (e.g., "Software Bus Network", "Time Services", "Executive Services", "Event Services", or "Table Services"). "Service" is intended to embrace all cFS functionality provided to software. "Service" within a network Quality of Service (QoS) model is a reasonable point of departure for the use of the term here.
- cFS middleware (like a network) exists between users while providing services. Like a network, cFS technical risks during execution can impact the timeliness, availability, accuracy or other attributes of provided services. Similarly, cFS programmatic risks can impact cost and schedule during development, test and operation.

Because cFS middleware generally resides between the OS and essential applications while not displaying cFS specific external goals, the cFS risk consequence during execution (e.g., performance risk impact, human safety risk impact, asset safety risk, mission success risk, etc.) could be estimated to be:

1. the most severe risk consequence of the loss or incorrect execution of the applications supported.

   This is a first order estimation that is relatively easily established but which excludes the risk consequence of emergent fault behaviors.

2. the most severe risk consequence of the loss or incorrect execution of multiple applications supported by cFS services and applications.

   This is a second order estimation and presumes that the middleware cannot cause a risk consequence greater than the supported applications' combined faults. This risk consequence estimation approach considers all supported applications and therefore correctly accounts for the shared "common mode fault" found within the supported applications as a group.

3. the most severe risk consequence of the loss or incorrect execution of each cFS service provided to each application.

   This could be considered the "real" cFS risk consequence, and an accurate expression of what the risk statements' express.

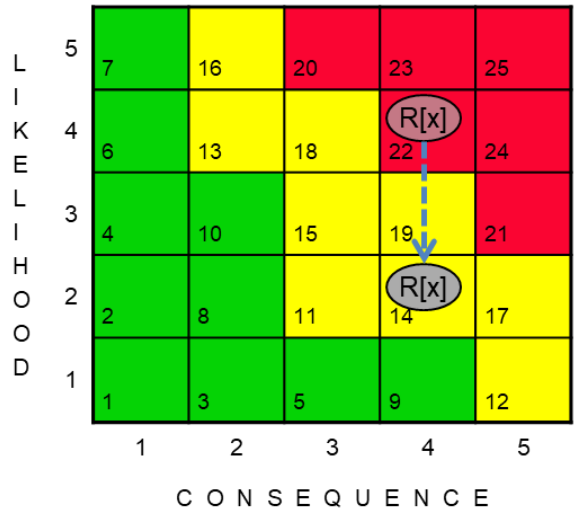
The cFS middleware risk consequence during development (schedule risk impact, cost risk impact, etc.) can be estimated in the same manner as any modified legacy SW code. Parameterized cost and schedule estimations based on the SLOC for the specific implementation could be higher than would be reasonable.


NASA

[SARP cFS V&V] Risk [2] – [External]

{Current Priority Score indicated by oval in Risk Matrix below; "[x]" represents [risk #] throughout.
If your score has changed, then show where the previous score was and where the new score is as shown in the example.}

Risk Title	cFS Verification and Validation Not Performed as Part of Some Identified Class A or B projects or Programs Use of cFS
Risk Statement	"Given that [Multiple projects or programs were observed applying cFS SW to Class A or Class B software systems without performing comprehensive verification and validation], there is a possibility of [flight experience reports not accurately reflecting software testing maturity in similar systems] adversely impacting [cFS software code including both executable code and supporting documentation (e.g. verification or validation evidence)], thereby leading to [greater than anticipated frequency or severity of cFS software failures or faults]."
Context Statement	NASA projects and programs have been observed to treat cFS software as a low risk Off The Shelf (OTS) software package citing prior projects' flight use. Projects appear unaware of the level of rigor applied to prior programs' SW preflight testing (verification, validation, or other cert. tests). Consequently subsequent mission or project specific cFS verification and validation has been reduced or eliminated in observed cases. This condition has been communicated to NASA Tech Authorities, but otherwise would be considered to deviate from the requirements of NPR 7150.2 Rev B e.g., *3.9.2 The project manager shall satisfy the following conditions when a COTS, GOTS, MOTS, or reused software component is acquired or used: [SWE-027]* *a. The requirements to be met by the software component are identified...* *e. The software component is verified and validated to the same level required to accept a similar developed software component for its intended use.* *... Note: The project responsible for procuring off-the-shelf software is responsible for documenting, prior to procurement, a plan for verifying and validating the software to the same level that would be required for a developed software component. ...*
Closure Criteria	Completion of verification and validation of the cFS instance in use for the individual mission, program or project.



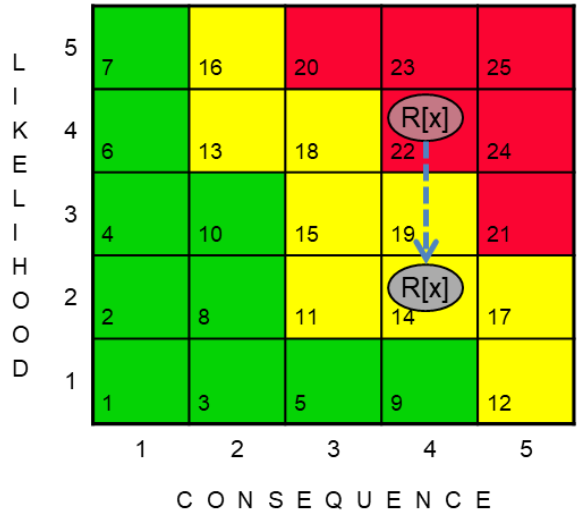
	Date
Sunrise	Critical Design Review
Sunset	Flight Readiness Review / Safety and Mission Success Review
Impact Horizon	[Program and project specific evaluation]

Consequence	Rationale	Likelihood	Rationale
[Program and project specific evaluation]	[Program and project specific evaluation]	[Program and project specific evaluation]	[Program and project specific evaluation]

[SARP cFS V&V] Risk [3] – [External]

{Current Priority Score indicated by oval in Risk Matrix below; "[x]" represent s [risk #] throughout.
If your score has changed, then show where the previous score was and where the new score is as shown in the example.}

Risk Title	Review of cFS Project Backlog Requires Access to an Access Controlled Repository
Risk Statement	Given that [Multiple projects or programs were observed applying cFS SW to Class A or Class B software systems without reviewing the backlog of reported software defects], there is a possibility of [unknown defects that impact project/program application exist] adversely impacting [the project's/program's defensive programing or operations to mitigate latent defects], thereby leading to [controllable defects being permitted to impact mission success or safety or uncontrolled defects risk not being properly understood by the user community].
Context Statement	Projects have been observed to omit the existing NASA requirement for review of OTS software defect lists. Two projects' points of contact were unaware that the backlog was available for review. If reviews are omitted and the omission is not submitted as a variance to the appropriate NPR 7150.2 Technical Authority, this would represent an omitted NASA requirement. *3.9.2 The project manager shall satisfy the following conditions when a COTS, GOTS, MOTS, or reused software component is acquired or used: [SWE-027]...* *f. The project has a plan to perform periodic assessments of vendor reported defects to ensure the defects do not impact the selected software components....*
Closure Criteria	Conformance to NASA expectations for reviewing OTS software defect reports and addressing each defect through: fault controls, error handling, operational work arounds, software changes to address defects, or management risk acceptance.



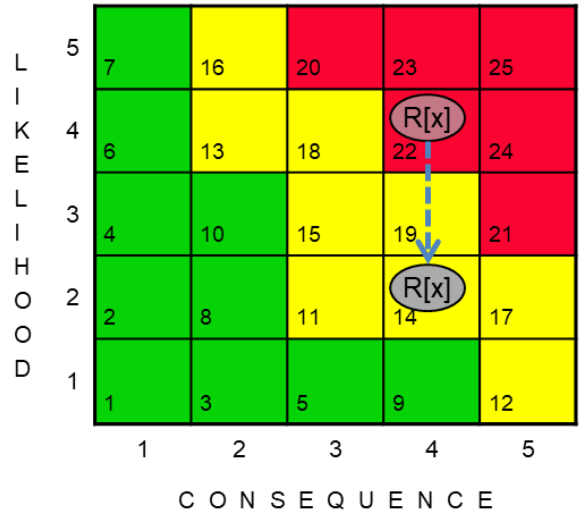
	Date
Sunrise	Preliminary Design Review
Sunset	System Retirement
Impact Horizon	[Program and project specific evaluation]

Consequence	Rationale	Likelihood	Rationale
[Program and project specific evaluation]	[Program and project specific evaluation]	[Program and project specific evaluation]	[Program and project specific evaluation]

[SARP cFS V&V] Risk [4] – [External]

{Current Priority Score indicated by oval in Risk Matrix below; "[x]" represent s [risk #] throughout.
If your score has changed, then show where the previous score was and where the new score is as shown in the example.}

Risk Title	Stability of the cFS Software
Risk Statement	Given that [cFS core and application software continues to mature and be applied across an increasing range of system (HW & OS SW) architectures while the user base remains relatively small (when compared to commercial products)], there is a possibility of [unidentified SW defects] adversely impacting [software reliability and functionality], thereby leading to [increased and misunderstood potential for the loss of system function, mission, assets, or personnel]."
Context Statement	cFS service and application software continues to mature, add capabilities, and resolve defects while operating as an open source software system selected by cost constrained NASA projects. User selections for operating system SW, interfacing application SW, interfacing HW and processing HW have been demonstrated to be wide ranging. Establishing and maintaining SW with proven flight success, improved maturity/reduced defect backlogs, satisfaction of user desires and low total program costs may not be possible. Reliability and safety may be adversely impacted, while obvious impacts to cost, user desires, and schedule may be overly considered due to those impacts being easier to track and prioritize.
Closure Criteria	Mature, low defect, and cost constrained total program costs that satisfy user desires/demands



	Date
Sunrise	Preliminary Design Review
Sunset	System Retirement
Impact Horizon	[Program and project specific evaluation]

Consequence	Rationale	Likelihood	Rationale
[Program and project specific evaluation]	[Program and project specific evaluation]	[Program and project specific evaluation]	[Program and project specific evaluation]