National Aeronautics and Space Administration

NASA Engineering and Safety Center Technical Bulletin No. 24-02

Considerations for Using Autonomous Flight Termination Software in Crewed Launch Vehicles

Autonomous flight termination systems (AFTS) are being progressively employed onboard launch vehicles to replace ground personnel and infrastructure needed to terminate flight or destruct the vehicle should an anomaly occur. This automation uses on-board real-time data and encoded logic to determine if the flight should be self-terminated. For uncrewed launch vehicles, FTS systems are required to protect the public and governed by the United States Space Force (USSF). For crewed missions, NASA must augment range AFTS requirements for crew safety and certify each flight according to human rating standards, thus adding unique requirements for reuse of software originally intended for uncrewed missions. This bulletin summarizes new information relating to AFTS to raise awareness of key distinctions, summarize considerations and outline best practices for incorporating AFTS into human-rated systems.

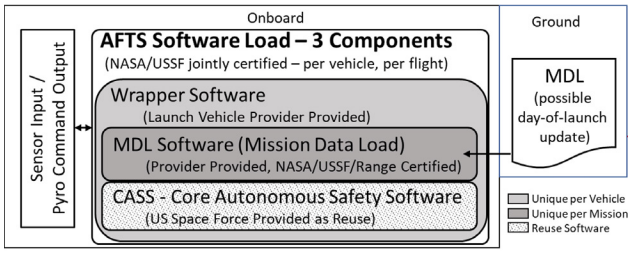
Key Distinctions - Crewed v. Uncrewed

There are inherent behavioral differences between uncrewed and crewed AFTS related to design philosophy and fault tolerance. Uncrewed AFTS generally favor fault tolerance against failure-to-destruct over failing silent in the presence of faults. This tenet permeates the design, even down to the software unit level. Uncrewed AFTS become zero-fault-to-destruct tolerant to many unrecoverable AFTS errors, whereas general single fault tolerance against vehicle destruct is required for crewed missions. Additionally, unique needs to delay destruction for crew escape, provide abort options and special rules, and assess human-in-the-loop insight, command, and/or override throughout a launch sequence must be considered and introduces additional requirements and integration complexities.

AFTS Software Architecture Components and Best-Practice Use Guidelines

A detailed study of the sole AFTS currently approved by USSF and utilized/planned for several launch vehicles was conducted to understand its characteristics, and any unique risk and mitigation techniques for effective human-rating reuse. While alternate software systems may be designed in the future, this summary focuses on an architecture employing the Core Autonomous Safety Software (CASS). Considerations herein are intended for extrapolation to future systems. Components of the AFTS software architecture are shown, consisting of the CASS, "Wrapper", and Mission Data Load (MDL) along with key characteristics and use guidelines. A more comprehensive description of each and recommendations for developmental use is found in Ref. 1.

Component	Characteristics	Use Guidelines
CASS	• Computes data from tracking sensors and estimates impact point • MDL rule execution • C++, evolving each release • GFE, controlled by USSF	• Work with USSF SLD 30 SEAE on acquisition and use • Treat as reuse software and fill gaps between RCC 319-19 and NPR 7150.2
Wrapper	• Unique per launch vehicle architecture, adapting software to vehicle-specific platform • Owned by launch vehicle provider or creator (possible reuse software)	• Implement features ensuring crew escape in all abort modes before the AFTS pyrotechnics fire • Implement two-stage commanding as needed • Handle CASS exceptions • Add telemetry for insight and independent monitoring
MDL	• Contains all flight termination decision logic • XML code written against USSF rsoML schema • Developed by launch provider and range, certified jointly between NASA, USSF range, and possible licensing authority	• Treat with equivalent rigor as comparable human decisions, reflecting that knowledge • Ensure all calculations are with valid data • Ensure tested and verified as safety-critical Class A • Validate on ground and onboard prior to use
Total	• Comparable in size and complexity to typical launch vehicle flight software	• Certify as a complete package



AFTS Software Architecture Components

Best Practices Certifying AFTS Software

Below are non-exhaustive guidelines to help achieve a human-rating certification for an AFTS.

AFTS Software Development and Certification Guidelines

• Develop per NPR7150.2 Class A Safety-Critical in addition to RCC 319-19
  • Treat CASS as reuse software and certify to level of intended use
  • Treat MDL as safety-critical software and fully test (i.e. MC/DC)
  • Accredit models and simulations used for AFTS verification

• Implement software fault-tolerance and prevention as required by human rating standards and summarized in Ref 6., for example:
  • Employ an independent monitor and/or a dissimilar backup (e.g. consider using the space vehicle's abort logic)
  • Employ crew/ground insight/control providing non-hazard time-to-effect
  • Detect and mitigate effects of erroneous software output

• Develop a Hazard Analysis including risks introduced by use of AFTS software along with associated mitigation controls, including treatment for:
  • Zero-fault tolerant design elements such as the MDL
  • Erroneous output software errors and software exception conditions
  • Failure conditions which lead directly to destruct, or lack of destruct

• Develop processes and controls to ensure integrity and validity of the AFTS software configuration, including the onboard MDL   • Ensure changes, including day-of-launch, are validated
• Employ "Test like you fly" with additional off nominal scenarios to fully exercise encoded decision logic within a flight-like computing environment

References

1. NASA/TP-20240009981: Best Practices and Considerations for Using Autonomous Flight Termination Software In Crewed Launch Vehicles https://ntrs.nasa.gov/citations/20240009981
2. "Launch Safety," 14 C.F.R., § 417 (2024).
3. NPR 8705.2C, Human-Rating Requirements for Space Systems, Jul 2017, nodis3.gsfc.nasa.gov/
4. NASA Software Engineering Requirements, NPR 7150.2D, Mar 2022, nodis3.gsfc.nasa.gov/
5. RCC 319-19 Flight Termination Systems Commonality Standard, White Sands, NM, June 2019.
6. "Considerations for Software Fault Prevention and Tolerance", NESC Technical Bulletin No. 23-06 https://ntrs.nasa.gov/citations/20230013383
7. "Safety Considerations when Repurposing Commercially Available Flight Termination Systems from Uncrewed to Crewed Launch Vehicles", NESC Technical Bulletin No. 23-02 https://ntrs.nasa.gov/citations/20230001890

NESC tech bulletin

For more information, contact Lorraine E. Prokop lorraine.e.prokop@nasa.gov.