Chapter 13:

The Application of System Safety to the Commercial Launch Industry

This chapter is intended for use as a pull-out handbook, separate from the FAA System Safety Handbook.

13.0 The Application of System Safety To the Commercial Launch Industry

13.1 Introduction

The office of the Associate Administrator for Commercial Space Transportation (AST), under Title 49, U.S. Code, Subtitle IX, Sections 70101-70119 (formerly the Commercial Space Launch Act), exercises the FAA's responsibility to:

> regulate the commercial space transportation industry, only to the extent necessary to ensure compliance with international obligations of the United State and to protect the public health and safety, safety of property, and national security and foreign policy interest of the United States, …encourage, facilitate, and promote commercial space launches by the private sector, ***recommend appropriate changes in Federal statutes, treaties, regulations, policies, plans, and procedures, and facilitate the strengthening and expansion of the United States space transportation infrastructure.*** [emphasis added]

The mandated mission of the AST is "…to protect the public health and safety and the safety of property…."

AST has issued licenses for commercial launches of both sub-orbital sounding rockets and orbital expendable launch vehicles. These launches have taken place from Cape Canaveral Air Station (CCAS), Florida, Vandenburg Air Force Base (VAFB), California, White Sands Missile Range (WSMR), New Mexico, Wallops Flight Facility (WFF), Wallops Island, Virginia, overseas, and the Pacific Ocean.

AST has also issued launch site operator licenses to Space Systems International (SSI) of California, the Spaceport Florida Authority (SFA), the Virginia Commercial Space Flight Authority (VCSFA), and the Alaska Aerospace Development Corporation (AADC). SSI operates the California Spaceport located on VAFB; SFA the Florida Space Port located on CCAS; VCSFA the Virginia Space Flight Center located on WFF; and AADC the Kodiak Launch Complex, located on Kodiak Island, Alaska.

13.2 Office of Commercial Space Transportation (AST)

AST is divided into three functional components, the office of the Associate Administrator (AST-1), the Space Systems Development Division (SSDD), and the Licensing and Safety Division (LASD).

13.2.1 The office of the Associate Administrator (AST-1)

AST-1 establishes policy, provides overall direction and guidance to ensures that the divisions function efficiently and effectively relative to the mandated mission "…to protect the public health and safety and the safety of property…."

13.2.2 The Space Systems Development Division (SSDD)

The SSDD assess new and improved launch vehicle technology and their impacts upon both the existing and planned space launch infrastructures. SSDD works with the FAA and DOD Air Traffic Services to ensure full integration of space transportation flights into the Space and Air Traffic Management System. SSDD is AST's interface with the Office of Science and Technology Policy (OSTP), other Government agencies, and the aerospace industry working to create a shared 2010 space launch operations vision and in the development of the Global Positioning (Satellite) System (GPS) for the guidance of launch vehicles and tracking at ranges. SSDD is also engaged in analyzes of orbital debris and its impact to current and future space launch missions and the commercialization of outer space.

13.2.3 The Licensing and Safety Division (LASD)

LASD's primary objective is to carry out AST's responsibility to ensure public health and safety through the licensing of commercial space launches and launch site operations, licensing the operation of non-Federal space launch sites, and determining insurance or other financial responsibility requirements for commercial launch activities. AST/LASD looks to ensure protection of public health and safety and the safety of property through its licensing and compliance monitoring processes.

13.3 LICENSING PROCESS

The components of the licensing process include a pre-licensing consultation period, policy review, payload review, safety evaluation, financial responsibility determination, and an environmental review. The licensing process components most concerned with the application of system safety methodologies are the ***safety evaluation***, ***financial responsibility determination***, and ***environmental determination***. A space launch vehicle requires the expenditure of enormous amounts of energy to develop the thrust and velocity necessary to put a payload into orbit. The accidental or inadvertent release of that energy could have equally enormous and catastrophic consequences, both near and far.

13.3.1 Safety Evaluation

It is the applicant's responsibility to demonstrate that they understand all hazards and risks posed by their launch operations and how they plan to mitigate them. Hazard mitigation may take the form of safety devices, protective systems, warning devices, or special procedures.

There are a number of technical analyses; some quantitative and some qualitative, that the applicant may perform in order to demonstrate that their commercial launch operations will pose no unacceptable threat to the public. The quantitative analyses tend to focus on 1) the reliability and functions of critical safety systems, and 2) the hazards associated with the hardware, and the risk those hazards pose to public property and individuals near the launch site and along the flight path, to satellites and other on-orbit spacecraft. The most common hazard analyses used for this purpose are Fault Tree Analysis, Failure Modes and Effects Analysis, and Over-flight Risk and On-Orbit Collision Risk analyses using the Poisson Probability Distribution. The qualitative analyses focus on the organizational attributes of the applicant such as launch safety policies and procedures, communications, qualifications of key individuals, and critical internal and external interfaces.

It is AST/LASD's responsibility to ensure that the hazard analyses presented by the applicant demonstrates effective management of accident risks by identifying and controlling the implicit as well as explicit hazards inherent in the launch vehicle and proposed mission. LASD must evaluate the applicant's safety data and safety related hardware/software elements and operations to ascertain that the demonstrations provided by the applicant are adequate and valid.
Specifically, the LASD evaluation is designed to determine if the applicant has:

- Identified all energy and toxic sources and implemented controls to preclude accidental or inadvertent release.
- Evaluated safety critical aspects, potential safety problems, and accident risk factors.
- Identified potential hazardous environments or events, and assessed their causes, possible effects and probable frequency of occurrence.
- Implemented effective hazard elimination, prevention or mitigation measures or techniques to minimize accident risk to acceptable levels.
- Specified the means by which hazard controls or mitigation methodology can be verified and validated.

13.3.2 Financial Responsibility Determination

Section 70112 of the Act requires that all commercial licensees demonstrate financial responsibility to compensate for the maximum probable loss from claims by:

- A third party for death, bodily injury, or property damage or loss resulting from an activity carried out under the license; and
- The U.S. Government against a person for damage or loss to government property resulting from an activity carried out under the license.

Section 70112 also requires that the Department of Transportation set the amounts of financial responsibility required of the licensee. The licensee can then elect to meet this requirement by:

- Proving it has financial reserves equal to or exceeding the amount specified, or
- Placing the required amount in escrow, or
- Purchasing liability insurance equal to the amount specified.

The most common and preferred method is via the purchase of liability insurance.

The methodology developed for setting financial responsibility requirements for commercial launch activities is called Maximum Probable Loss (MPL) analysis[1]. MPL analysis was developed to protect launch participants from the maximum probable loss due to claims by third parties and the loss of government property during commercial launch activities. Note that this is maximum probable loss, not maximum possible loss. Generally speaking, MPL is determined by identifying all possible accident scenarios, examining those with the highest potential losses for both government property and third party, and then estimating the level of loss that would not be exceeded at a given probability threshold. If the launch is to take place from a private licensed range and no government property is at risk, no government property financial responsibility requirement will be issued.

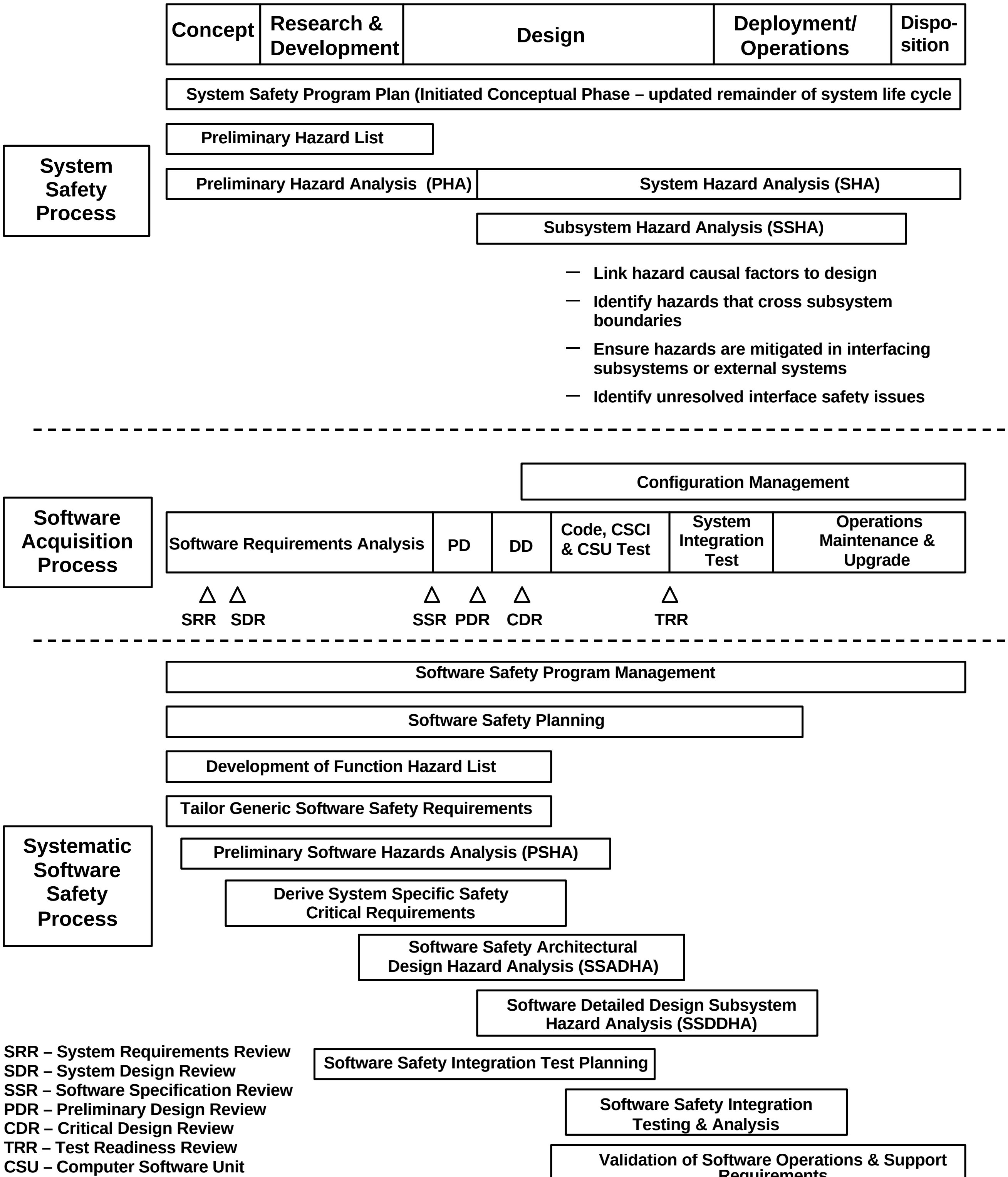
An integral part of, and critical input to the MPL, is the Facility Damage and Personnel (DAMP) Injury Analysis[2]: DAMP uses information about launch vehicles, trajectories, failure responses, facilities and populations in the launch area to estimate the risk and casualty expectations from impacting inert debris, secondary debris and overpressures from impact explosions. Together, the MPL and DAMP analyses are used to determine the financial responsibility determinations necessary to insure compensation for losses resulting from an activity carried out under the commercial license.

13.3.3 Environmental Determination

The environmental determination ensures that proposed commercial space launch activities pose no threat to the natural environment. The National Environmental Policy Act (NEPA) of 1969, as amended, requires that: Federal agencies consider the environmental consequences of major Federal actions; take actions that protect, restore, and enhance the environment; and ensure that environmental information is available to public officials and citizens before making decisions and taking action. The licensing of commercial space launch activities, either for a launch or launch site, is considered a major Federal action. Consequently, AST is responsible for analyzing the environmental impacts associated with proposed commercial space launch activities. AST is also responsible for the assessing the applicant’s preparation and submittal of Environmental Assessments and Environmental Impact Statements to ensure compliance with the NEPA.

[1] Futron Corporation developed the MPL Analysis methodology employed by AST.
[2] Research Triangle Institute developed the DAMP Analysis methodology employed by AST.

Figure 13-3
System Safety, Software Acquisition and Systematic Software Acquisition Process


Concept
Research & Development
Design
Deployment/ Operations
Dispo-sition
System Safety Process
System Safety Program Plan (Initiated Conceptual Phase – updated remainder of system life cycle
Preliminary Hazard List
Preliminary Hazard Analysis (PHA)
System Hazard Analysis (SHA)
Subsystem Hazard Analysis (SSHA)
– Link hazard causal factors to design
– Identify hazards that cross subsystem boundaries
– Ensure hazards are mitigated in interfacing subsystems or external systems
– Identify unresolved interface safety issues
Software Acquisition Process
Configuration Management
Software Requirements Analysis
PD
DD
Code, CSCI & CSU Test
System Integration Test
Operations Maintenance & Upgrade
SRR
SDR
SSR
PDR
CDR
TRR
Systematic Software Safety Process
Software Safety Program Management
Software Safety Planning
Development of Function Hazard List
Tailor Generic Software Safety Requirements
Preliminary Software Hazards Analysis (PSHA)
Derive System Specific Safety Critical Requirements
Software Safety Architectural Design Hazard Analysis (SSADHA)
Software Detailed Design Subsystem Hazard Analysis (SSDDHA)
Software Safety Integration Test Planning
Software Safety Integration Testing & Analysis
Validation of Software Operations & Support Requirements

SRR – System Requirements Review
SDR – System Design Review
SSR – Software Specification Review
PDR – Preliminary Design Review
CDR – Critical Design Review
TRR – Test Readiness Review
CSU – Computer Software Unit
CSCI – Computer Software Configuration Item
PD – Preliminary Design
DD – Detailed Design

13.4 SYSTEM SAFETY ENGINEERING PROCESS

13.4.1 Overview

The System Safety Engineering Process is the structured application of system safety engineering and management principles, criteria, and techniques to address safety within the constraints of operational effectiveness, time, and cost throughout all phases of a system's life cycle. The intent of the System Safety Engineering Process is to identify, eliminate, or control hazards to acceptable levels of risk throughout a system's life cycle.

This process is performed by the vehicle developer/operator. Because of the complexity and variety of vehicle concepts and operations, such a process can help ensure that all elements affecting public safety are considered and addressed. Without such a process, very detailed requirements would have to be imposed on all systems and operations, to ensure that all hazards have been addressed which could have the undesired effect of restricting design alternatives and innovation or could effectively dictate design and operations concepts.

The process (as described in Mil Std 882C) includes a System Safety Program Plan (SSPP). The SSPP (or its equivalent) provides a description of the strategy by which recognized and accepted safety standards and requirements, including organizational responsibilities, resources, methods of accomplishment, milestones, and levels of effort, are to be tailored and integrated with other system engineering functions. The SSPP lays out a disciplined, systematic methodology that ensures all risks – all events and system failures (probability and consequence) that contribute to expected casualty – are identified and eliminated, or that their probability of occurrence is reduced to acceptable levels of risk.

The SSPP should indicate the methods employed for identifying hazards, such as Preliminary Hazards Analysis (PHA), Subsystem Hazard Analysis (SSHA), Failure Mode and Effects Analysis (FMEA), Fault Tree Analysis. Risk Mitigation Measures are likewise identified in the plan. These include avoidance, design/redesign, process/procedures and operational rules and constraints.

The System Safety Engineering Process identifies the safety critical systems. Safety critical systems are defined as any system or subsystem whose performance or reliability can affect public health and safety and safety of property. Such systems, whether they directly or indirectly affect the flight of the vehicle, may or may not be critical depending on other factors such as flight path and vehicle ability to reach populated areas. For this reason, it is important to analyze each system for each phase of the vehicle mission from ground operations and launch through reentry and landing operations. Examples of potentially safety critical systems that may be identified through the system safety analysis process using PHA or other hazard analysis techniques may include, but are not limited to:

- Structure/integrity of main structure
- Thermal Protection System (e.g., ablative coating)
- Temperature Control System (if needed to control environment for other critical systems)
- Main Propulsion System
- Propellant Tanks
- Power Systems
- Propellant Dumping System

- Landing Systems
- Reentry Propulsion System
- Guidance, Navigation and Control System(s), Critical Avionics (Hardware and Software) - includes Attitude, Thrust and Aerodynamic Control Systems
- Health Monitoring System (hardware and software)
- Flight Safety System (FSS)
- Flight Dynamics (ascent and reentry) for stability (including separation dynamics) and maneuverability
- Ground Based Flight Safety Systems (if any) including telemetry, tracking and command and control systems
- Depending on the concept, additional "systems" might include pilot and life support systems and landing systems if they materially affect public health and safety
- Others identified through hazard analysis

13.4.2 Validation of Safety Critical Systems

Through the system safety process, the applicant demonstrates that the proposed vehicle design and operations satisfy regulatory requirements and that the system is capable of surviving and performing safely in all operating environments including launch, orbit, reentry and recovery. Documentation must show adequate design, proper assembly, and vehicle control during all flight phases. Documentation is expected to consist of design information and drawings, analyses, test reports, previous program experience, and quality assurance plans and records.

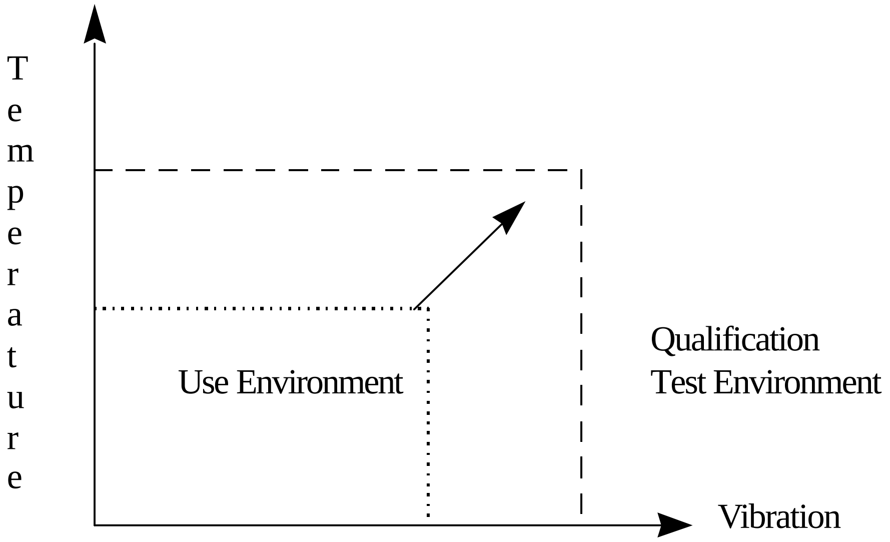
AST uses a pre-application consultation process to help a potential applicant to understand what must be documented and to help identify potential issues with an applicant's proposed activities that could preclude its obtaining a license. The pre-application process should be initiated by the applicant early in their system development (if possible during the operations concept definition phase) and maintained until their formal license application is completed. This pre-application process should be used to provide AST with an understanding of the safety processes to be used, the safety critical systems identified, analysis and test plan development, analysis and test results, operations planning and flight rules development.

Analyses may be acceptable as the primary validation methodology in those instances where the flight regime cannot be simulated by tests, provided there is appropriate technical rationale and justification.

Qualification tests, as referenced in the safety demonstration process and the System Safety Program Plan, are normally conducted to environments higher than expected. For example, expendable launch vehicle (ELV) Flight Safety Systems (FSS) are qualified to environments a factor of two or higher than expected. (See Figure 13-2) These tests are conducted to demonstrate performance and adequate design margins and may be in the form of multi-environmental ground tests, tests to failure, and special flight tests. Such tests are normally preceded with detailed test plans and followed by test reports.[3]

[3] Test plans are important elements of the ground and flight test programs. Such plans define, in advance, the nature of the test (what is being tested and what the test is intended to demonstrate with respect to system functioning, system performance and system reliability). The test plan should be consistent with the claims and purpose of the test and wherever appropriate, depending on the purpose of the test, clearly defined criteria for pass and fail should be identified. A well-defined test plan and accompanying test report may replace observation by the FAA.

Figure 13-2: Relationship of Use Environment to Qualification Test



In addition, Quality assurance (QA) records are useful in establishing verification of both design adequacy and vehicle assembly and checkout (workmanship).

Table 13-1, Validation Acceptance Matrix, identifies sample approaches that may be employed to validate acceptance for critical systems. Examples of types of analyses, ground tests, and flight tests are provided following this matrix. (Note: Quality Assurance programs and associated records are essential where analysis or testing, covering all critical systems, are involved.)

Table 13-1: Validation Acceptance Matrix

Candidate Critical System	Analyses	Ground Test	Flight Test
Structure/Integrity of Main Structure	X	X	P
Thermal Protection	X	P	P
Environmental Control (temp, humidity)	X	X	X
Propulsion: Main, Auxiliary and Reentry (de-orbit)	X	P	P
Propellant Tank Pressurization	X	X	P
GN&C, Critical Avionics *; includes de-orbit targeting (e.g., star-tracker, GPS)	X	X	X
Health Monitoring *	X	X	X
Flight Safety System (FSS)*	X	X	X
Recovery and Landing*	X	P	P
Ordnance* (other than Safety)	X	X	X
Electrical and Power*	X	X	X
Telemetry and Tracking and Command*	X	X	X
Flight Control (ascent, separation, reentry) *	X	X	X
FSS Ground Support Equipment (if any) *	X	X	N/A

P - partial; cannot satisfy all aspects

X **-** If in sufficient detail when combined with test results or selected analyses
* **-** Includes both hardware and software

13.4.3 Analyses

There are various types of analyses that may be appropriate to help validate the viability of a critical system or component. The following provides examples of some types of critical systems analysis methodologies and tools.

- Mechanical Structures and Components (Vehicle Structure, Pressurization, Propulsion System including engine frame thrust points, Ground Support Equipment)
- Types of Analyses: Structural Loads, Thermal, Fracture Mechanics, Fatigue, Form Fit & Function
- Software Tools for Analyses: Nastran, Algor, Computational Fluid Dynamics codes, CAD/CAM
- Thermal Protection System
- Types of Analyses (for TPS and Bonding Material): Transient and Steady State Temperature Analyses, Heat Load, and Heating and Ablative Analyses.
- Software Tools for Analyses: SINDA by Network Analysis Inc.
- Electrical/Electronic Systems & Components (Electrical, Guidance, Tracking, Telemetry, Navigation, Communication, FSS, Ordnance, Flight Control and Recovery)
- Types of Analyses: Reliability, FMEA, Single Failure Point, Sneak Circuit, Fault Tree, Functional Analysis, Plume effects
- Software Tools for Analyses: MathCad, Relex, and FaultrEase
- Propulsion Systems (Propulsion, FSS, Ordnance, Flight Control)
- Types of Analyses: Analytical Simulation of nominal launch and abort sequences for Main Engines, Orbital Maneuvering System (including restart for reentry-burn) and Attitude Control System; capacity analysis for consumables; Plume Flow Field Modeling
- Software Tools for Analyses: Nastran, Algor, SPF-III, and SINDA
- Aerodynamics (Structure, Thermal, Recovery)
- Types of Analyses: Lift, Drag, Stability, Heating, Performance, Dispersion, Plume effects
- Software Tools for Analyses: Post 3/6 DOF, Computational Fluid Dynamics Codes Monte Carlo Simulation Codes
- Software (Guidance, Tracking & Telemetry & Command, FSS, Flight Control and Recovery)
- Types of Analyses: Fault Tree, Fault Tolerance, Software Safety (including abort logic), Voting Protocol Dead Code, Loops, and Unnecessary Code
- Validation Methodologies, such as ISO 9000-3[4]

[4] ISO 9000-3 is used in the design, development, and maintenance of software. Its purpose is to help produce software products that meet the customers' needs and expectations. It does so by explaining how to control the quality of both products and the processes that produce these products. For software product quality, the standard highlights four measures: specification, code reviews, software testing and measurements.

13.4.4 Ground Test

Ground tests include all testing and inspections performed by the applicant prior to flight, including qualification, acceptance and system testing. It is anticipated that an applicant will perform various types of ground tests to validate the capability of critical systems and components. The following provides examples of some types of critical systems validation ground tests. Again these are *only examples* and should not be construed as the only types of ground tests which may be used to validate a specific system for a specific operational environment, nor should it be interpreted that all of these example ground tests will be necessary to validate a specific system.

Mechanical Systems and Components (Vehicle Structure, Pressurization, Propulsion System including engine frame thrust points, Ground Support Equipment)

> ***Types of Tests:*** Load, Vibration (dynamic and modal), Shock, Thermal, Acoustic, Hydro-static, Pressure, Leak, Fatigue, X-ray, Center of Gravity, Mass Properties, Moment of Inertia, Static Firing, Bruceton Ordnance, Balance, Test to Failure (simulating non-nominal flight conditions), Non-Destructive Inspections

Electrical/Electronic Systems (Electrical, Guidance, Tracking, Telemetry and Command, Flight Safety System (FSS), Ordnance, Flight Control and Recovery)

> ***Types of Tests:*** Functional, Power/Frequency Deviation, Thermal Vacuum, Vibration, Shock, Acceleration, X-ray, recovery under component failures, abort simulations, TDRSS integration testing (up to and including pre-launch testing with flight vehicle)

Propulsion Systems (Propulsion, FSS, Ordnance, Flight Control)

> ***Types of Tests:*** Simulation of nominal launch and abort sequences for engines (including restart, if applicable), Orbital Maneuvering System (including restart for reentry-burn) and Attitude Control System; Environmental testing (Thermal, Vibration, Shock, etc.)

Thermal Protection System

> ***Types of Tests*** (for TPS and bonding material): Thermal, Vibration, Humidity, Vacuum, Shock

Aerodynamics (Structure, Thermal, Recovery)

> ***Types of Tests:*** Wind Tunnel, Arc Jet, Drop Tests (Landing Systems)

Software (Electrical, Guidance, Tracking, Telemetry, Command, FSS, Ordnance, Flight Control and Recovery)

> ***Types of Tests:*** Functional, Fault Tolerance, Cycle Time, Simulation, Fault Response, Independent Verification and Validation, Timing, Voting Protocol, Abort sequences (flight and in-orbit) under non-nominal conditions with multiple system failures, Integrated Systems Tests

13.4.5 Flight Tests

If an applicant's System Safety Plan includes a flight test program, then a considerable amount of planning is needed to define the flight test program that will establish the performance capabilities of the vehicle for routine and repetitive commercial operations. When flight testing is indicated, a flight test plan will be needed to demonstrate that the vehicle's proposed method of operations is acceptable and will not be a hazard to the public health and safety, and safety of property.

The purpose of flight-testing is to verify the system performance, validate the design, identify system deficiencies, and demonstrate safe operations. Experience repeatedly shows that while necessary and important, analyses and ground tests cannot and do not uncover all potential safety issues associated with new launch systems. Even in circumstances where all known/identified safety critical functions can be exercised and validated on the ground, there is still the remaining concern with unrecognized or unknown interactions (“the unknown unknowns”).

The structure of the test program will identify the flight test framework and test objectives, establish the duration and extent of testing; identify the vehicle’s critical systems, identify the data to be collected, and detail planned responses to nominal and unsatisfactory test results.

Test flight information includes verification of stability, controllability, and the proper functioning of the vehicle components throughout the planned sequence of events for the flight. All critical flight parameters should be recorded during flight. A post-flight comparative analysis of predicted versus actual test flight data is a crucial tool in validating safety critical performance. Below are examples of items from each test flight that may be needed to verify the safety of a reusable launch vehicle. Listed with each item are examples of what test-flight data should be monitored or recorded during the flight and assessed post-flight:

Vehicle/stage launch phase: Stability and controllability during powered phase of flight.

- Vehicle stage individual rocket motor ignition timing, updates on propellant flow rates, chamber temperature, chamber pressure, and burn duration, mixture ratio, thrust, specific impulse (ISP)
- Vehicle stage trajectory data (vehicle position, velocity, altitudes and attitude rates, roll, pitch, yaw attitudes)
- Vehicle stage Attitude, Guidance and Control system activities
- Functional performance of the Vehicle Health Monitoring System
- Functional performance of the Flight Safety System/Safe Abort System
- Electrical power, and other critical consumables, usage and reserves (i.e. gases, fluids, etc…)
- Actual thermal and vibroacoustic environment
- Actual structural loads environment

Staging/separation phase of boost and upper stages: Stable shutdown of engines, and nominal separation of the booster & upper stages.

- Separation activity (timestamp, i.e., separation shock loads, and dynamics between stamps)
- Functional performance of the Vehicle Health Monitoring System
- Electrical power, and other critical consumables, usage and reserves (i.e. gases, fluids, etc…)
- Functional performance of the Flight Safety System/Safe Abort System

Booster stage turn-around (re-orientation) or “loft” maneuver phase (if applicable):

- Rocket motor re-start (if applicable): timing, updates on propellant flow rates, chamber temperature, chamber pressure, burn duration, mixture ratio, thrust, ISP
- Attitude, Guidance and Control system activities
- Actual structural loads environment
- Actual thermal and vibroacoustic environment

- Functional performance of the Flight Safety System/Safe Abort System

Booster stage flyback phase (if applicable): Flyback engine cut-off, fuel dump or vent (if required), nominal descent to the planned impact area, proper functioning and reliability of the RLV landing systems.

- Booster stage post-separation (flyback) trajectory data
- Electrical power usage and reserves
- Booster stage landing system deployment activity (timestamp)
- Actual thermal and vibroacoustic environment
- Actual structural loads environment
- Functional performance of the Vehicle Health Monitoring System
- Functional performance of the Flight Safety System/Safe Abort System
- Attitude, Guidance and Control system activities

Vehicle stage ascent phase (if multistage): nominal ignition of the stage's engine, stability and controllability of the stage during engine operation, orbital insertion – simulated (for suborbital) or actual – of the vehicle.

- Vehicle individual rocket motor ignition timing, updates on propellant flow rates, chamber temperature, chamber pressure, and burn duration
- Vehicle circularization and phasing burn activities (ignition timing, updates on propellant flow rates, chamber temperature, chamber pressure, and burn duration)
- Vehicle trajectory data (vehicle position, altitude, velocity, roll, pitch, yaw attitudes at a minimum)
- Attitude, guidance and control system activities
- Functional performance of the Vehicle Health Monitoring System
- Functional performance of the Flight Safety System/Safe Abort System
- Electrical power, and other critical consumables, usage and reserves (i.e. gases, fluids, etc…)
- Actual structural loads environment
- Actual thermal and vibroacoustic environment

Vehicle descent (including vehicle's de-orbit burn targeting and execution phases): Function of the programmed flight of the vehicle/upper stage to maintain the capability to land (if reusable) at the planned landing site, or to reenter for disposal (if expendable), assurance of fuel dump or depletion, and proper descent and navigation to the planned or alternate landing site.

- Vehicle pre-deorbit burn trajectory data
- Vehicle deorbit burn data (ignition timing, updates on propellant flow rate, chamber temperature, chamber pressure, and burn duration)
- Vehicle descent trajectory data (position, velocity, and attitude)
- Attitude, Guidance and Control system activities

- Actual thermal and vibroacoustic environment
- Actual structural loads environment
- Functional performance of the Vehicle Health Monitoring System
- Functional performance of the Flight Safety System/Safe Abort System
- Electrical power and other critical consumables usage and reserves (i.e. gases, fluids, etc…)
- Vehicle landing system deployment activity (timestamp)

13.4.6 Performance and Reliability Data

Performance and reliability data may be supported by flight history on other vehicles with similar or comparable safety critical systems, sub-systems, and components, and by conducting both analyses and tests, at the respective levels. A flight history could mean extensive documentation might not be required if it can be shown through test results, analyses, or empirical data, that the flight regimes experienced are similar to the proposed flight regime. The degree of applicability of data depends on the degree of similarity to environmental conditions and how environmental conditions compare to the history and anticipated reactions of this system. Even when the same system, sub-system, or component is known to have an extensive (and favorable) flight history in the same or more severe environments, interfaces and integration with other systems must still be examined and tested. Another method of acquiring data is through estimating system, sub-system, and component 3-sigma performance and reliability numbers from testing evaluations and (where applicable) flight data.

The use of similarity is not new to launch operations. EWR 127-1, Paragraph. 4.14.1.2, states: as required, qualification by similarity analysis shall be performed; if qualification by similarity is not approved, then qualification testing shall be performed. For example, if component A is to be considered as a candidate for qualification by similarity to a component B that has already been qualified for use, component A shall have to be a minor variation of component B. Dissimilarities shall require understanding and evaluation in terms of weight, mechanical configuration, thermal effects, and dynamic response. Also, the environments encountered by component B during its qualification or flight history shall have to be equal to or more severe than the qualification environments intended for component A.

13.4.7 Operational Controls

There is an interrelationship between the system design capabilities and the systems operational limitations. Figure 2 depicts the relationship between the vehicle systems and the scope of operations within which the vehicle is operated. What constitutes a safety critical system may depend on the scope and nature of the vehicle design and its proposed operations. Intended operational requirements affect the proposed vehicle design requirements and vehicle capabilities/limitations and also establish the operational system constraints necessary to protect public health and safety. For example, reusable launch vehicle landing sites may have to be within some minimum cross-range distance from the orbital ground trace because of cross-range limitations of the vehicle. A vehicle operator may choose, or be required, to mitigate certain vehicle limitations through the use of operational controls rather than relieving vehicle limitations through design changes.

Test parameters and analytic assumptions will further define the limits of flight operations. The scope of the analyses and environmental tests, for example, will constitute the dimensions of the applicant's demonstration process and therefore define the limits of approved operations if a license is issued. Such testing limits, identified system and subsystem limits, and analyses also are expected to be reflected in mission monitoring and mission rules addressing such aspects as commit to launch, flight abort, and commit to reentry.

Vehicle capabilities/limitations and operational factors such as launch location and flight path each affect public risk. The completion of system operation demonstrations, such as flight simulations and controlled flight tests, provide additional confidence in the vehicle systems and performance capabilities. As confidence in the systems overall operational safety performance increases, key operational constraints such as restrictions on overflight of populated areas may be relaxed.

The following are examples of the types of operations-related considerations that may need to be addressed by the applicant when establishing their operations scenarios.

Launch commit criteria/rules

Human override capability to initiate safe abort during launch and reentry

System monitoring, inspection and checkout procedures

For re-flight: inspection and maintenance

Selected primary and alternate landing sites for each stage

Surveillance/control of landing areas

Standard limits on weather

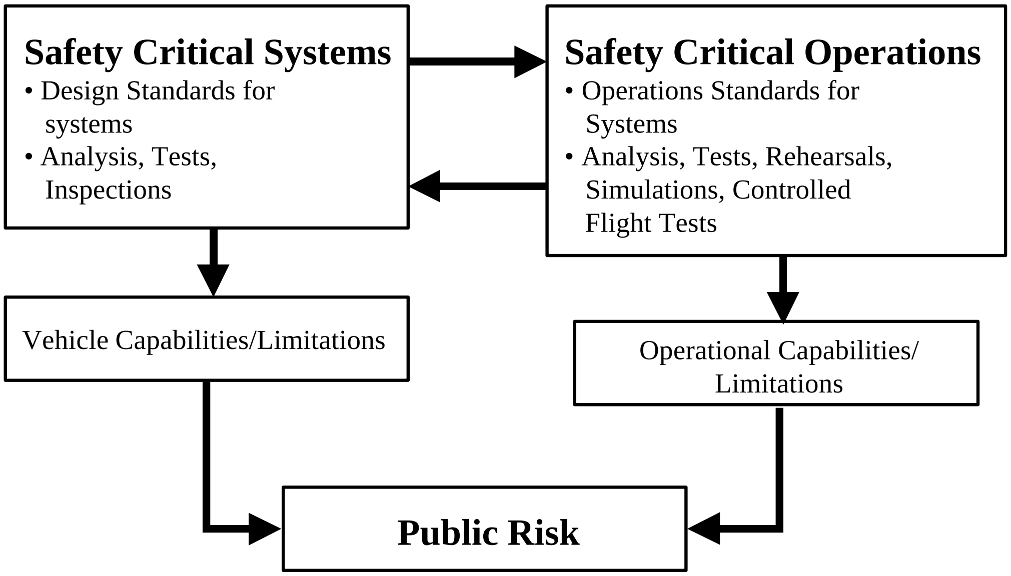
Coordination with appropriate air space authorities

Limits on flight regime (ties in with analysis, testing and demonstrating confidence in system performance and reliability)

Limits on over-fight of populated areas

Others identified through hazard analysis

Figure 13-2: Interrelationship between Safety Critical Systems and Safety Critical Operations


Safety Critical Systems
• Design Standards for systems
• Analysis, Tests, Inspections
Safety Critical Operations
• Operations Standards for Systems
• Analysis, Tests, Rehearsals, Simulations, Controlled Flight Tests
Vehicle Capabilities/Limitations
Operational Capabilities/ Limitations
Public Risk

13.4.8 Determination of Risk to the Public

Expected casualty is used in the space transportation industry as a measure of risk to public safety. Expected casualty is the expected average number of human casualties per mission. Human casualty is defined as a fatality or serious injury. The application of the expected casualty analysis to determine public risk is further defined in FAA Advisory Circular 431-02.

13.4.9 Determination of Need for Additional Risk Mitigation

The results of the expected casualty analysis may identify the need for additional risk mitigation measures that need to be employed. These measures may include additional operational controls or may require the redesign of certain safety critical systems. These additional risk mitigation measures would be evaluated within the System Safety Process and the resultant risk to the public would be determined.

13.5 SOFTWARE SAFETY

13.5.1 Safety Critical Software

Safety-critical software plays an ever-increasing role in Commercial Space Transportation (CST) computer systems. To preserve CST flight integrity, software-based hazards must be identified and eliminated or reduced to acceptable levels of risk. Particular concern surrounds potential software-induced accidents occurring during CST launch and reentry. Due to mission complexity, software failures manifested at these critical times can cause serious accidents. Populated areas would suffer major harm if defective software were to permit CST vehicles to violate their defined safety launch limits. Safety-critical software, relative to CST launch vehicles, payloads and ground support equipment is inherently defined as any software within a control system containing one or more hazardous or safety critical functions. Safety critical functions are usually but not always associated with safety-critical systems. Therefore, the following definition for safety –critical systems may also be applied to safety-critical functions. A safety-critical system (or function) has been inherently defined as any system or subsystem (or function) whose performance or reliability can affect (i.e. malfunction or failure will endanger) public health, safety and safety of property.[5]

13.5.2 Systematic Software Safety Process

Introduction

The Systematic Software Safety Process (SSSP) encompasses the application of an organized periodic review and assessment of safety-critical software and software associated with safety-critical system, subsystems and functions. The Systematic Software Safety Process consist primarily of the following elements:

- Software safety planning
- The software safety organization
- A software safety team
- Application of the software safety process during all life cycle phases
- Identification and application of life cycle phase-independent software safety activities
- Identification of special provisions
- Software safety documentation

Software Safety Planning

Software system safety planning is deemed essential early in the software life cycle. Most importantly, planning should impose provisions for accommodating safety well before each of the software design, coding, testing, deployment and maintenance phases starts in the cycle. Moreover, these provisions are to be planned carefully to impact minimally the software development process. The software system safety plan should contain provisions assuring that:

- Software safety organization is properly chartered and a safety team is commissioned in time.
- Acceptable levels of software risk are defined consistently with risks defined for the entire system.
- Interfaces between software and the rest of the system’s functions are clearly delineated and understood.
- Software application concepts are examined to identify safety-critical software functions for hazards.
- Requirements and specifications are examined for safety hazards (e.g. identification of hazardous commands, processing limits, sequence of events, timing constraints, failure tolerance, etc.)
- Design and implementation is properly incorporated into the software safety requirements.
- Appropriate verification and validation requirements are established to assure proper implementation of software system safety requirements.
- Test plans and procedures can achieve the intent of the software safety verification requirements.

[5] Reference D.

- Results of software safety verification efforts are satisfactory.

The Institute of Electrical and Electronic Engineering (IEEE) offers a comprehensive standard (Standard for Software Safety Plans) focusing solely on planning. The Standard articulates in sufficient detail both software safety management and supporting analyses. The Standard's annex describes the kind of analyses to be performed during the software requirements, design, code, test and change phases of the traditional life cycle. Similar planning models are provided by the Department of Defense (DOD) Defense Standard 00-55-Annex B.

Software Safety Organization

Safety oversight consists of a staff function crossing several organizational boundaries. By its nature, it is meant to interact with other staff and line functions, including program or project management, software system design, quality assurance, programming, reliability, testing, human factors, and operations. Accountability-wise, the ultimate responsibility for the development and operation of a safe software system(s) rests with the CST applicant or licensed operator. Thus, the applicant's or operator's top management should be committed to supporting the software safety process across all these staff and line functions.

A software safety organization can take one of many shapes, depending on the needs of the applicant or licensed operator. However, the following requisites are recommended:

- Centralized authority and responsibility dedicated to the safety initiatives
- Safety team independence, and
- High enough safety team status relative to the rest of the organization.

Centralization allows a single organization to focus entirely on hazards and their resolutions during any life cycle phase, be it design, coding or testing. Independence prevents bias and conflicts of interest during organizationally sensitive hazard assessment and management. A high status empowers the team to conduct its mission with sufficient visibility and importance. By endorsing these requisites, CST applicants and operators will indicate they are attentive to the safety aspects of their project or mission.

Software Safety Team

Safety planning also calls for creating a software safety team. Team size and shape depends commensurately on mission size and importance. To be effective, the team should consist of analytical individuals with a sufficient system engineering background. Military Standard (MIL STD) 882C provides a comprehensive matrix of minimum qualifications for key system safety personnel. It can apply to software system safety as well, provided professional backgrounds include sufficient experience with software development (software requirements, design, coding, testing, etc.)

Several typical activities expected of the team range from identifying software-based hazards to tracing safety requirements and limitations in the actual code, to developing software safety test plans and reviewing test results for their compliance with safety requirements.

Software Safety During Life Cycle Phases

The SSSP should support a structured program life cycle model that incorporates both the system design and engineering, and software acquisition process. Prominent software life cycle models include the waterfall and spiral methodologies. Although different models may carry different lifecycle emphasis, the adopted model should not affect the SSSP itself. For discussion purposes only, this enclosure adopts a waterfall model (subject to IEEE/IEA Standard for Information Technology-software life cycle processes No. 12207.) For brevity, only some phases (development, operation, maintenance and support) of the Standard are addressed in terms of their relationship to software safety activities. This relationship is summarized in Table 13-2 The table's contents partly reflect some of the guidance offered by the National Aeronautics and Space Administration (NASA) Standard 8719.13A and NASA Guidebook GB-1740.13-96.

Table 13-2: Software Safety Activities Relative to the Software Life Cycle

Life Cycle Phase	**Corresponding** **Safety Activity**	**Inputs**	**Expected Results**	**Milestones To Be Met**
Concept/ **Requirements/** **Specifications**	-Review software concept for safety provisions -Derive generic and system-specific software safety requirements. -Analyze software requirements for hazards. -Identify potential software/system interface hazards -Develop Functional Hazards List (FHL) -Develop initial Preliminary Software Hazard Analysis (PSHA)	-Preliminary Hazard Analysis (PHA) [from system safety analysis] -Generic and system-wide safety specs.	PSHA Report PHL	Software Concept Review (SCR) Software Requirements Review (SRR) and Software Specification Review (SSR)
Architecture/ **Preliminary** **Software Design**	At high design level: -Identify Safety Critical Computer Software Components (SCCSCs) -Verify correctness & completeness of architecture -Ensure test coverage of software safety requirements.	PSHA	Software Safety Architectural Design Hazard Analysis (SSADHA) Report	Preliminary Design Review (PDR)
Detailed Design	At the low design(unit) level: -Focus on SCCSCs at the unit level. -Verify correctness/ completeness of detail. Design	PSHA SSADHA	Software Safety Detailed Design Hazard Analysis (SSDDHA) Report	Critical Design Review (CDR)
Implementation **Coding**	-Examine correctness & completeness of code from safety requirements. -Identify possibly unsafe code. -Walk-through/audit the code	PSHA, SSADHA, SSDDHA	Software Safety Implementation Hazard Analysis (SSIHA) report	Test Readiness Review (TRR)
Integration and **Testing**	-Ensure test coverage of software safety requirements. -Review test documents and results for safety requirements. -Final SSHA	Test documents	-Software Safety Integration Testing (SSIT) Report -Final SSHA report	Acceptance
Operations and **Maintenance**	-Operating and Support Hazard Analysis (O&SHA)	All of the above plus all incidents reports	O&SHA Report(s), as required	Deployment

Figure 3 provides a composite overview of the entire safety process. The figure consists of three parts. The top part reflects the broader System Safety Process described in draft Advisory Circular 431.35-2. The middle part illustrates a typical waterfall Software Acquisition Process life cycle. The bottom part also partly corresponds to the Systematic Software Safety Process. In Figure 3, all processes shown in horizontal bars are subject to a hypothetical schedule with time duration not drawn to any scale.

Phase-independent software safety activities

NASA's Software Safety Standard 8719.13A mentions activities not tied to specific phases. The Standard lists the following ones meant to occur throughout the life cycle:

- Tracing safety requirements keeping track of the software safety requirements during design, coding and testing, including the correspondence between these requirements and the system hazard information.
- Tracking discrepancies between the safety and development aspects of the software.
- Tracking changes made to the software to see if they impact the safety process.
- Conducting safety program reviews to verify if safety controls are being implemented to minimize hazards.

Special Provisions

Commercial Off the Shelf (COTS): COTS software targets a broad range of applications, with no specific one envisioned ahead of time. Therefore, care must be taken to ensure COTS software presence minimizes risk when it becomes embedded or coupled to specific applications. Consideration ought to be given to designing the system such that COTS software remains isolated from safety-critical functions. If isolation is not possible, then safeguards and oversight should be applied.

Software Reuse: Reusable software originates from a previous or different application. Usually, developers intend to apply it to their current system, integrating it "as is" or with some minor modifications. The Software Safety Team verification/validation plan, etc.) Annex B should serve as a general model for preparing software safety documents

The results of most of the safety analyses activities usually require preparing several hazard analysis reports documenting the findings of the safety team. The team has also the responsibility of presenting their findings to decision-making management at critical milestones, like the Software Requirements Review (SRR), Preliminary Design Review (SDR), Critical Design Review (CDR), etc. Towards this end, DOD Defense Standard 00-55-Annex E describes how to prepare a software safety "case". The Standard defines a case as "a well-organized and reasoned justification, based on objective evidence, that the software does or will satisfy the safety aspects of the Software Requirement".

13.5.3 Software Safety Documentation

Numerous documents are to be prepared and distributed during the SSSP. To track them, a comprehensive checklist, such as that cited in MIL STD 882 may be applied. Two highly recommended documents are the Safety Assessment Report (SAR) and the System Safety Hazard Analysis Report (SSHA). DOD Defense Standard 00-55-Annex B offers a detailed outline of the contents of numerous software safety documents (software safety plan, case or report, records log, audit plan, audit report, quality plan, risk management plan, verification/validation plan, etc.) Annex B should serve as a general model for preparing software safety documents. The results of most of the safety analyses activities usually require preparing several hazard analysis reports documenting the findings of the safety team.

The team also has the responsibility of presenting their findings to decision-making management at critical milestones, like the Software Requirements Review (SRR), Preliminary Design Review (SDR), Critical Design Review (CDR), etc. Towards this end, DOD Defense Standard 00-55-Annex E describes how to prepare a software safety "case". The Standard defines a case as "a well-organized and reasoned justification, based on objective evidence, that the software does or will satisfy the safety aspects of the Software Requirement".

13.5.4 Safety Critical Software functions

Software can be labeled defective if it does not perform as expected. Major classes of defects are:

- Software not executing
- Software executing too late, too early, suddenly or out of sequence, or
- Software executing but producing wrong information.

In turn, defective software can be labeled hazardous if it consists of safety-critical functions that command, control and monitor sensitive CST systems. Some typical software functions considered safety-critical include:

- Ignition Control: any function that controls or directly influences the pre-arming, arming, release, launch, or detonation of a CST launch system.
- Flight Control: any function that determines, controls, or directs the instantaneous flight path of a CST vehicle.
- Navigation: any function that determines and controls the navigational direction of a CST vehicle.
- Monitoring: any function that monitors the state of CST systems for purposes of ensuring its safety.
- Hazard Sensing: any function that senses hazards and/or displays information concerning the protection of the CST system.
- Energy Control: any function that controls or regulates energy sources in the CST system.
- Fault Detection: any function that detects, prioritizes, or restores software faults with corrective logic.
- Interrupt Processing: any function that provides interrupt priority schemes and routines to enable or disable software-processing interrupts.
- Autonomous Control: any function that has autonomous control over safety-critical hardware.
- Safety Information Display: any function that generates and displays the status of safety-critical hardware or software systems.
- Computation: any function that computes safety-critical data.

13.5.5 Software Safety Risk and Hazard Analysis

Risk Assessment

A key element in system safety program planning is the identification of the acceptable level of risk for the system. The basis for this is the identification of hazards. Various methodologies used in the identification of hazards are addressed in Sections 2.3 & 2.4 of draft AC 431.35-2. Once the hazards and risks are identified, they need to be prioritized and categorize so that resources can be allocated to the functional areas having an unacceptable risk potential. Risk assessment and the use of a Hazard Risk Index (HRI) Matrix as a standardized means with which to group hazards by risk are described in Attachment 2, Sections 6.1 & 6.2 of draft AC 431.35-2. This section presents specialized methods of analyzing hazards, which possess software influence or causal factors and supplements the HRI presented in draft AC 431.35-2.

The Hazard Risk Index presented in draft AC 431.35-2 is predicated on the probability of hazard occurrence and the ability to obtain component reliability information from engineering sources. Hardware reliability modeling of a system is well established; however, there is no uniform, accurate or practical approach to predicting and measuring the software reliability portion of the system. Since software does not fail in the same manner as hardware, in that it is not a physical entity, it does not wear out, break, or degrade over time; software problems are referred to as a software error. Software errors general occur due to implementation or human failure mechanisms (such as documentation errors, coding errors, incorrect interpretation of design requirements, specification oversight, etc.) or requirement errors (failure to anticipate a set of conditions that lead to a hazard). Unlike hardware, software has many more failure paths than hardware, making it difficult to test all paths. Thus the ultimate goal of software system safety is to find and eliminate the built-in unintended and undesired hazardous functions driven by software in a CST system.

Classification of Software Safety Risk

There are two basic steps in classifying safety risk for software. The first being the establishment of severity within the context of the CST system and then applying an acceptable methodology for determining the software's influence on system level risks. Refer to Figures 13-4 and 13-5. Regardless of the contributory factors (hardware, software, or human error) the severity of risk as present in draft AC 431.35-2 Attachment 2, Section 6.1.2, Figure 6.1.2, remain applicable criteria for the determination of hazard criticality for those risks possessing software contributory factors.

The second half of the equation for the classification of risk is applying an acceptable methodology for determining the software's influence on system level hazards. The probability factors contained in draft AC 431.35-2 has been determined for hardware based upon historical "best" practices. Data for the assignment of accurate probabilities to software error has not matured. Thus alternate methods for determining probability propagated by software causal factors need to be used. Numerous methods of determining software effects on hardware have been developed and two of the most commonly used are presented in MIL-STD 882C and RTCA DO-178 and are shown in Figure 4. These methods address the software's "control capability" within the context of the software casual factors. An applicant Software System Safety Team should review these lists and tailor them to meet the objectives of their CST system and integrated software development program.

This activity of categorizing software causal factors is for determining both likelihood, and the design, coding, and test activities required to mitigate the potential software contributor. A Software Hazard

Criticality (SHC) Matrix, similar to the hazard risk index (HRI)[6] matrix is used to determine the acceptability of risk for software hazards. Figure 3 shows an example of a typical SHC matrix using the control categories of MIL-STD 882C [Mil882C]. The SHC matrix can assist the software system safety team in allocating software safety requirements against resources and in the prioritization of software design and programming tasks.

Software Hazard Analysis/Risk Mitigation

Fault tree analysis (FTA) may be used to trace system-specific software safety-critical functional hazards[7]. The hazard software causal factor nodes are then traced to the appropriate mitigating CST System Requirement, design segment, and coding segment. The hazard should be tracked through the test procedure development; to assure the test procedures have been written sufficiently to demonstrate the hazard is adequately controlled (mitigated).

Software Safety Analysis Methods and Tools[8]

The following is not intended to be an all-inclusive or exhaustive list of software safety analysis methods and tools; nor does it represent an explicit or implicit AST recommendation thereof.

[6] See Attachment 2, Section 6.2 of AC 431.35-2 for discussion and illustration of HRI.

[7] The actual analysis techniques used to identify hazards, their causes and effects, hazard elimination, or risk reduction requirements and how they should be met should be addressed in the applicant's System Safety Program Plan. The System Safety Society's System Safety Handbook identifies additional system safety analysis techniques that can be used.

[8] Reference E

MIL-STD 882C	RTCA-DO-178B
(I) Software exercises autonomous control over potentially hazardous hardware systems, subsystems or components without the possibility of intervention to preclude the occurrence of a hazard. Failure of the software or a failure to prevent an event leads directly to a hazard's occurrence. II(a) Software exercises control over potentially hazardous hardware systems, subsystems, or components allowing time for intervention by independent safety systems to mitigate the hazard. However, these systems by themselves are not considered adequate. II(b) Software item displays information requiring immediate operator action to mitigate a hazard. Software failure will allow or fail to prevent the hazard's occurrence. III(a) Software items issues commands over potentially hazardous hardware systems, subsystem, or components requiring human action to complete the control function. There are several, redundant, independent safety measures for each hazardous event. III(b) Software generates information of a safety critical nature used to make safety critical decisions. There are several, redundant, independent safety measures for each hazardous event. (IV) Software does not control safety critical hardware systems, subsystems, or components and does not provide safety critical information.	(A) Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a catastrophic failure condition for the vehicle. (B) Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a hazardous/severe major failure condition of the vehicle. (C) Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a major failure condition for the vehicle. (D) Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of system function resulting in a minor failure condition for the aircraft. (E) Software whose anomalous behavior, as shown by the system safety assessment process, would cause or contribute to a failure of function with no effect on vehicle operational capability or pilot workload. Once software has been confirmed as level E by the certification authority, no further guidelines of this document apply.

Figure 13-3: Software Hazard Criticality Matrix*

CONTROL CATEGORY	CATASTROPHIC	CRITICAL	MARGINAL	NEGLIGIBLE
(I) S/W without possibility of intervention– leads directly to hazard occurrence	1	1	3	5
(IIa) S/W with time for intervention– can not stand alone	1	2	4	5
(IIb) S/W displays information but requires operator to mitigate hazard - allow or fail to prevent hazard occurrence.	1	2	4	5
(IIIa) S/W issues commands requiring human action to complete control function– several redundant, independent measures for each event.	2	3	5	5
(IIIb) S/W generate information of a safety critical nature to make safety critical decisions - several redundant, independent measures for each event.	2	3	5	5
(IV) S/W does not control safety critical H/W systems or provide safety-critical information	3	4	5	5

1	High Risk	Significant Analyses and Testing Resources
2	Medium Risk	Requirements and Design Analysis and Dept Test Required
3	Moderate Risk	High Levels of Analysis and Testing Acceptable with Managing Activity Approva
4	Moderate Risk	High Levels of Analysis and Testing Acceptable with Managing Activity Approv
5	Low Risk	Acceptable

*Extracted from MIL-STD 882C

It is intended to provide a limited representative sampling of those software safety analysis methods and tools available to the CST licensee or operator. General systems safety analysis have been omitted in that they are addressed in Paragraph 4.3. It is the licensee or operator's responsibility to assess the applicability and viability of a particular analysis method or tool to their CST, methods of operations, and organizational capabilities.

- Code Inspection: a formal review process during which a safety team checks the actual code, comparing it stepwise to a list of hazard concerns.
- Hardware/Software Safety Analysis[9]: this analysis is a derivative of the system PHA[10]. The PHA when integrated with the requirements leveled upon the software will identify those programs, routines, or modules that are critical to system safety and must be examined in depth.
- Software Failure Modes and Effects Analysis (SFMEA)[11]: identifies software related design deficiencies through analysis of process flow-charting. It also identifies interest areas for verification /validation and test and evaluation. Technique is used during and after the development of software specifications. The results of the PHA and SSHA, if complete, can be used ass a guide for focusing the analysis.
- Software Fault Tree Analysis (SFTA)[12]: used to identify the root cause(s) of a "top" undesired event. When a branch of the hardware FTA leads to the software of the system, the SFTA is applied to that portion of software controlling that branch of the hardware FTA. The outputs from the SFMEA, Software Requirements Hazard Analysis (SRHA), Interface Analysis, and Human Factors/Man-Machine Interface Analysis can provide inputs to the SFTA. SFTA can be performed at any or all levels of system design and development.
- Software Hazard Analysis (SHA)[13]: used to identify, evaluate, and eliminate or mitigate software hazards by means of a structured analytical approach that is integrated into the software development process.
- Software Sneak Circuit Analysis (SSCA)[14]: is used to uncover program logic that could cause undesired program outputs or inhibits, or incorrect sequencing/timing. When software controls a safety critical event, an SSCA can help detect a condition that would cause a catastrophic mishap if the cause were an inadvertent enabling condition.

Generic Software Safety Provisions

Two recommended sources for the applicant of generic software safety provisions used in the design and development of CST systems that have safety-critical applications are the Joint Software System Safety Committee Software System Safety Handbook and Eastern and Western Range Safety Requirements, (EWR 127-1). Using the generic software safety provision previously discussed and other available software safety "best practices" the applicant should be able to develop system software safety requirements. This should be done early in the software engineering process, in order for software design features to be specified that will eliminate, mitigate, or control hazards/risks at an acceptable level with minimal program impact.

[9] Alternate Names: Software Hazard Analysis (SHA) and Follow-On Software Hazard Analysis.

[10] See Paragraph 4.3.

[11] Alternate Names: Also knows as Software Fault Hazard Analysis (SFHA) and Software Hazardous Effects Analysis (SHEA).

[12] Alternate Name: Also know as Soft Tree Analysis (STA).

[13] Alternate Name: Software Safety Analysis (SSA).

[14] Should be cross-referenced to system SCA.

Design and Development Process Guidelines

The following guidelines should be applied to the software design and development process:

- A software quality assurance program should be established for systems having safety-critical functions.
- At least two people should be thoroughly familiar with the design, coding, testing and operation of each software module in the CST system.
- The software should be analyzed throughout the design, development, and maintenance processes by a software system safety team to verify and validate the safety design requirements have been correctly and completely implemented.
- The processes as described in the software development plan should be enforceable and auditable. Specific coding standards or testing strategies should be enforced and they should be independently audited.
- Desk audits, peer reviews, static and dynamic analysis tools and techniques, and debugging tools should be used to verify implementation of identified safety-critical computing system functions.

System Design Requirements and Guidelines

The following system design requirements and guidelines should apply:

- The CST system should have at least one safe state identified for each operation phase.
- Software should return hardware systems under the control of software to a designed safe state when unsafe conditions are detected.
- Where practical, safety-critical functions should be performed on a standalone computer. If this is not practical, safety-critical functions should be isolated to the maximum extent practical from non-critical functions.
- Personnel not associated with the original design team should design the CST system and its software for ease of maintenance.
- The software should be designed to detect safety-critical failures in external hardware input or output hardware devices and revert to a safe state upon their occurrence.
- The software should make provisions for logging all system errors detected.
- Software control of safety-critical functions should have feedback mechanisms that give positive indications of the function’s occurrence.
- The system and software should be designed to ensure that design safety requirements are not violated under peak load conditions.
- Applicant should clearly identify an overall policy for error handling. Specific error detection and recovery situations should be identified.
- When redundancy is used to reduce the vulnerability of a software system to a single mechanical or logic failure, the additional failure modes from the redundancy scheme should be identified and mitigated.
- The CST system should be designed to ensure that the system is in a safe state during power-up.

- The CST system should not enter an unsafe or hazardous state after an intermittent power transient or fluctuation.
- The CST system should gracefully degrade to a secondary mode of operation or shutdown in the event of a total power loss so that potentially unsafe states are not created.
- The CST system should be designed such that a failure of the primary control computer will be detected and the CST system returned to a safe state.
- The software should be designed to perform a system level check at power-up to verify that the system is safe and functioning properly prior to application of power to safety-critical functions.
- When read-only memories are used, positive measures, such as operational software instructions, should be taken to ensure that the data is not corrupted or destroyed.
- Periodic checks of memory, instruction, and data buss(es) should be performed.
- Fault detection and isolation programs should be written for safety-critical subsystems of the computing system.
- Operational checks of testable safety-critical system elements should be made immediately prior to performance of a related safety-critical operation.
- The software should be designed to prevent unauthorized system or subsystem interaction from initiating or sustaining a safety-critical sequence.
- The system design should prevent unauthorized or inadvertent access to or modification of the software and object coding.
- The executive program or operating system should ensure the integrity of data or programs loaded into memory prior to their execution.
- The executive program or operating system should ensure the integrity of data and program during operational reconfiguration.
- Safety-critical computing system functions and their interfaces to safety-critical hardware should be controlled at all times. The interfaces should be monitored to ensure that erroneous or spurious data does not adversely affect the system, that interface failures are detected, and that the state of the interface is safe during power-up, power fluctuations & interruptions, in the event of system errors or hardware failure.
- Safety-critical operator display legends and other interface functions should be clear, concise and unambiguous and, where possible, be duplicated using separate display devices.
- The software should be capable of detecting improper operator entries or sequences of entries or operations and prevent execution of safety-critical functions as a result.
- The system should alert the operator to an erroneous entry or operation.
- Alerts should be designed such that routine alerts are readily distinguished from safety-critical alerts.
- Safety-critical computing system functions should have one and only one possible path leading to their execution.
- Files used to store safety-critical data should be unique and should have a single purpose.

- The software should be annotated, designed, and documented for ease of analysis, maintenance, and testing of future changes to the software. Safety-critical variables should be identified in such a manner that they can be readily distinguished from non-safety-critical variables.

Configuration Control

The overall System Configuration Management Plan should provide for the establishment of a Software Configuration Control Board (SCCB) prior to the establishment of the initial baseline. The SCCB should review and approve all software changes (modifications and updates) occurring after the initial baseline is been established.

The software system safety program plan should provide for a thorough configuration management process that includes version identification, access control, change audits, and the ability to restore previous revisions of the system.

Modified software or firmware should be clearly identified with the version of the modification, including configuration control information. Both physical and electronic “fingerprinting” of the version are encouraged.

Testing

Systematic and thorough testing should provide evidence for critical software assurance. Software test results should be analyzed to identify potential safety anomalies that may occur. The applicant should use independent test planning, execution, and review for critical software. Software system testing should exercise a realistic sample of expected operational inputs. Software testing should include boundary, out-of-bounds and boundary crossing test conditions. At a minimum, software testing should include minimum and maximum input data rates in worst case configurations to determine the system capabilities and responses to these conditions. Software testing should include duration stress testing. The stress test time should be continued for at least the maximum expected operation time for the system. Testing should be conducted under simulated operational environments. Software qualification and acceptance testing should be conducted for safety-critical functions.

<u>References</u>:

AST Licensing And Safety Division Directive No. 001, Licensing Process and Procedures dated March 15, 1996.

FAA Advisory Circular AC 431-01, Reusable Launch Vehicle System Safety Process, dated April 1999 (Draft)

Code of Federal Regulations, Commercial Space Transportation, Department of Transportation Title 14, Federal Aviation Administration, Chapter III, Part 415 – Launch Licenses, and Part 431 – Launch and Reentry of a Reusable Launch Vehicle (RLV)

FAA Advisory Circular AC 431-03, Software System Safety (Draft)

System Safety Society, System Safety Handbook, 2nd Edition, dated July 1997

Joint Software System Safety Committee Software System Safety Handbook

Eastern and Western Range Safety Requirements, EWR 127-1.

The Application of System Safety to the Commercial Launch Industry Licensing Process, FAA/ASY Safety Risk Assessment News Reports No. 97-4 and 97-5