22 July 24

MEMORANDUM FOR RECORD

To: Office of the Under Secretary of Defense for Research and Engineering (OUSD(R&E))
From: DoD Joint Weapon Safety Working Group (JWSWG)

Subj: Endorsement of White Paper Guidance to Perform Modes, States, and Transitions Hazard Analysis, 31 August 2023

Ref: (a) DoD Instruction 5000.69 "DoD Joint Services Weapon and Laser System Safety Review Processes" of 9 Nov 2011
(b) Military Standard 882E "Department of Defense Standard Practice for System Safety", 11 May 2012

Encl: (1) White Paper Guidance to Perform Modes, States and Transitions Hazard Analysis, 31 August 2023

Reference (a) establishes policy and assigns responsibilities for the Department of Defense (DoD) Joint Services Weapon and Laser System Safety Review Processes. Reference (a) requires Joint Service safety reviews for weapon and laser systems that will be used by two or more DoD Components and establishes a DoD Joint Weapon Safety Working Group (JWSWG) that will coordinate and liaise with the DoD Laser System Safety Working Group (LSSWG) on joint safety review processes and policy. Additionally, reference (a) authorizes publication of supporting guidance to provide specific information on the DoD Joint Services weapon and laser system safety processes.

Fully assessing the safety of a system with complex behavior is an especially difficult task. Viewing such systems as a set of states and state transitions is very useful to characterize their complex behavior. Viewing a system as a set of states and modes and as transitions between states and modes is very useful for describing complex behaviors. Understanding state transitions is part of system analysis and design. Systems with complex behavior and many subsystem interactions require an analysis technique that can organize the complexity into subcategories of behavior along logical lines. The traditional FHA, as described in reference (b), is a foundational System Safety Engineering (SSE) analysis in a System Safety Program and is one of the most important analyses that the system safety analyst will perform. What the FHA alone is not keyed to do is to examine variant behavior within complex software systems. To accomplish this, safety engineers need an analysis technique that can organize the complexity into subcategories of behavior along logical lines. This new guidance provided in enclosure (1) describes how to perform a Modes, States and Hazard Analysis (MSTHA). This document is being released to provide specific guidance to the DoD community on performing this special type of software safety analysis.

Subj: Endorsement of White Paper Guidance to Perform Modes, States, and Transitions Hazard Analysis, 31 August 2023

The Modes and States Transition Hazard Analysis guidance provided in enclosure (1) is endorsed by the undersigned as Co-Chairs of the DoD JWSWG. Based on this endorsement, the DoD JWSWG Co-Chairs recommend formal processing and issuance of enclosure (1).

MCCREARY.SHAWNA.M.1230236190 Digitally signed by MCCREARY.SHAWNA.M.1230236190 Date: 2024.08.02 14:51:08 -04'00'

Ms. Shawna M. McCreary (SES)
(Navy, WSESRB Chair),
Co-Chair, DoD Joint Weapon
Safety Working Group

[illegible] Digitally signed by HAMILTON.IAN.T.1257726874 Date: 2024.07.31 14:06:22 -04'00'

Mr. Ian T. Hamilton (SSTM)
(Army, AWSRB Chair),
Co-Chair, DoD Joint Weapon
Safety Working Group

LAZAR.ANDREW.T.1235651544 Digitally signed by LAZAR.ANDREW.T.1235651544 Date: 2024.08.02 11:55:05 -06'00'

Col Andrew T. Lazar
(Air Force, NNMSRB Chair),
Co-Chair, DoD Joint Weapon
Safety Working Group

White Paper

GUIDANCE TO PERFORM MODES, STATES AND TRANSITIONS HAZARD ANALYSIS

22 June 2024

POC: Michael H. Demmick, michael.h.demmick.civ@us.navy.mil

Accreditations

Michael H Demmick
Executive Secretary JWSWG
NOSSA N31E
Indian Head, MD

Curt H Danhauser
Lead Engineer
Booz Allen Hamilton
Arlington, VA

Stuart Whitford
Senior Lead Engineer
Booz Allen Hamilton
Arlington, VA

Robert Alex
Lead Engineer
Booz Allen Hamilton
Arlington, VA

GUIDANCE TO PERFORM
MODES, STATES AND TRANSITIONS HAZARD ANALYSIS

<u>FORWARD:</u> Fully assessing the safety of a system with complex behavior is an especially difficult task. Viewing such systems as a set of states and state transitions is very useful to characterize their complex behavior. To accomplish this, an analytical technique is needed to facilitate organizing the complexity into subcategories of behavior along logical lines. This paper will describe how to perform a Modes, States and Hazard Analysis (MSTHA). This analysis proceeds by reviewing the system design artifacts and identifying existing state diagrams for the major system components with any safety significance. The Functional Hazard Analysis (FHA) is reviewed for any functions that pertain to the state behavior of the system. Safety states in the diagram (those in which safety-significant functions are executed) are identified and all transitions into (guards) and out of (fail-safes) each safety state are gathered. Transition functions are then traced into the FHA to identify which ones do not exist in the FHA. Each safety state is examined within the overall system context in an attempt to glean any guard/fail-safe functions that are missing. Further analysis is done to ensure that functions dealing with system data used in state transition logic are traced to hazards. New hazards are then documented associated with any insufficient safety fault tolerance such as missing guard and fail-safe state transition functions and safety data verification functions. Lastly, a check is made to verify a strong implementation of state transition logic for safety behavior which includes periodic safety checks of correct state.

1. REFERENCES

a. DoD Manual 3800.AB, "DoD Unmanned System Safety Engineering Precepts Guide for DoD Acquisition." August 2021.
b. Military Standard 882E, "System Safety." May 11, 2012.
c. JSSSEH, "Joint Software Systems Safety Engineering Handbook, Version 1.0" August 27, 2010.
d. Allied Ordnance Publication - 52, " Guidance On Software Safety Design And Assessment Of Munition-Related Computing Systems." November 2008.

2. Modes, States and Hazard Analysis (MSTHA) Process

<u>Introduction:</u>
Viewing a system as a set of states and transitions between states is very useful for describing complex behaviors. Understanding state transitions is part of system analysis and design. Systems with complex behavior and many subsystem interactions require an analysis technique that can organize the complexity into subcategories of behavior along logical lines. This analysis is a useful check on the FHA as it looks at particular aspects of a system that the FHA isn't keyed to do - namely variant behavior. The DoD Unmanned System Safety Engineering Precepts Guide for DoD

Acquisition (ref. a) states in paragraph 6.1.4, "a states and modes analysis should be focused on the system safety risks associated with the operational modes of a system, the designed states which a system or subsystem can be in during each mode, and the transitions between those modes or states."

Implementing a state concept to control complex behavior in a system is much preferred over simply placing logical statements where certain conditions are checked before allowing certain actions. Recall the concept of magic numbers in software, where rather than using literal values like 100, 0.5, 2500 or 55% in code, it is better to create named constant variables to hold important numbers. This allows the numbers to be changed in one place if that value were to need updating. Similarly, it is very helpful to collect complex logic in the form of named states which represent particular sets of parameter values. If separate logic is implemented at various places within the source code whenever a determination is needed before calling a safety-significant function, then it is possible that some but not all instances of logic could be modified such that perhaps only some but not all parameters precluding energizing a certain device are checked in one instance but not another. Thus, one part of the code (perhaps the power control subsystem) would be in an effective state where the radar can be energized, and another (perhaps the operator GUI) would not allow RADAR ON. For severely hazardous systems, this is unacceptable as the system as a whole is not in any one state and perhaps some sections of code are in an invalid state. Best practices suggest implementing state control logic in one place and allowing various parts of the operating software access to the single variable that represents the system's operational state for purposes of allowing and disallowing safety significant functionality. When a state machine is implemented correctly a variable representing the element state can be initialized to a valid first state and thereafter the element state variable only ever changes by one of the transition functions which only ever changes the state variable from a valid current state to another valid destination state. The element state variable thus is never set to an invalid state.

States and Modes Terminology

The terms state, mode, phase, etc. have come to mean different things in different contexts throughout the engineering discipline. Unmanned System Safety Engineering Precepts Guide provides useful guidance regarding such terms. The guide states that the term mode "identifies operational segments within the system mission" while the term state, "identifies the conditions in which a system or subsystem can exist" and is "a subset of a mode." In addition, "[a] system or subsystem may be in only one state at a time." The guide further states that "[t]he overall safety of a system depends upon understanding its states within various modes and during transitions between them; this is particularly true in UMSs [Unmanned Systems]."

For our purposes, we will adhere to that terminology. Thus, a state will be used to refer to state behavior of the individual elements or major components within a larger system. State behavior is mutually exclusive such that an element can be in only one state at a time and that there are defined transition mechanisms that allow the element to move from one state to another. Examples of elements or major system components are radar systems, INS, fuzes, Electronic Safe and Arm Devices (ESAD), weapon launchers, guns, hatches, lasers, deployable wings, electable equipment such as dome covers or aero wedges, data links, target trackers. Also, the navigation systems for air/ground/surface /subsurface/space vehicles will also employ state behavior.

Systems often have a concept of employment mode which maps to the different environment or purpose to which the system is used. Examples include Training Mode, Maintenance Mode, Flight Test Mode, Captive Carry or Operational Mode. Often the system is physically changed or modified to support different modes of operation, such as explosives removed for flight test or captive carry modes, or networks may be disconnected when in Training Mode. Due to the need to qualify all software builds that are in fleet use, often systems will use the same software build for each of these uses with some method being employed to allow the software to discover at runtime which mode it is in. The term Employment Mode is used in this paper. Note that with modes, there may not be any ability to transition between modes or only between some modes (such as Operational and Training), with the system being brought up in one mode and then shutdown after each event.

Note that transitions between states of system elements are implemented by software. The proper safety design and implementation of software state transition logic in safety systems will be the focus of this paper. Modes such as Employment Mode can definitely affect element state behaviors, but it is the element behavior which controls the release of energy which can potentially lead to hazards.

Beginning the Analysis

This paper describes what specific steps should be done as part of a states and modes hazard analysis. For clarity, the steps are described by use of several tables, but a single spreadsheet or set of linked spreadsheets could also be used to facilitate the analysis and the collection of findings. The actual implementation details are immaterial to this discussion as long as the system knowledge is collected and analyzed. This guidance focuses on what must be done, and not how to accomplish it. However accomplished, the states and modes hazard analysis results should be clearly supported by specific and complete data and notes.

As specified in the MIL-STD-882E (ref. b) and expounded upon in the JSSSEH (ref. c), detailed software analysis and derivation of UML diagrams including state diagrams falls under the heading of safety design analysis and are specified as tasks to be accomplished for software requiring LOR-1 and LOR-2 level tasking. This includes a states and modes hazard analysis. This analysis can of course also be employed for systems with LOR 3 or LOR 4 functionality as well.

Step 1. Understand the System. The system engineer (SE) when performing this analysis should first review system description documentation to understand the system behavior and investigate any state design. This includes reviewing DoDAF OV-1, CONOPS, system requirements, software requirements, system design documents and any existing hazard analyses such as PHL, PHA, SHA, and SSHA.

Step 1 proceeds with a review of all pertinent design artifacts to gain an understanding of how the system functions in the various modes of operation and employment and especially how the various system elements exhibit variant behavior depending on particular sets of system parameters that are operative at the different times along the mission phases of the system.

Step 2. Document the State and Modes Behavior of the System. In this next step, the specifics of the system's variant behavior are captured, delineated, diagrammed, and documented. The SE collects any State machine diagrams from system design artifacts and if needed, creates a state diagram from descriptions of behavior within system requirements and design documents. This step proceeds for the elements within the system that are in any way associated with hazards. If a system element has variant behavior depending on what the system is doing as it progresses its mission, then state behavior exists whether the original designers created a state diagram or not. It is vital to diagram the state behavior of each system element to facilitate this analysis.

Transition diagrams are collected and created for safety-significant system elements and also for any moded behavior, especially if those defined modes govern the execution of safety-significant system functions.

Step 3. List System Functions Associated With Variant Behavior. Next, the analyst looks at the functions within the FHA and flags those functions that pertain to variant behavior or that map to states and modes illustrated in the system's element state transition diagrams. This process is an examination of all the functions in the FHA that refer to transitions into and out of the various states or modes. Also, the SE will note any functions that mention logic such as "if the vehicle is above a minimum altitude, arm the weapon" or "if the system fails BIT, then turn off the radar." These might indicate state behavior. We will be especially interested in the transitions into and out of safety-significant states. From analysis of all of these flagged functions, we may identify some logic that hints at state behavior that is not already reflected in the element state diagrams. This is one of the strengths of this analysis – the identification of all state behavior. It is important to note that often a system's FHA may be too high level and may not be detailed enough to obtain the functions needed for further state behavior analysis. In this case, the system software requirements documents should be examined to capture the needed functionality. If the software is particularly safety critical, it will be necessary to create a low-level FHA for the system - since one will be necessary for a complete safety program in any event. In that case, the states and modes analysis will wait and be performed once an appropriate FHA is available. A system engineer cannot assume that all aspects of system design – especially state machine design – has been done thoroughly. Note that transitions into or out of a system state can be complex. For instance, entrance into a weapon system's ARMED state may require a check of seven different parameters which are all necessary. This means that effectively these seven parameters or conditions are logically AND-ed together in the guard transition function since all must be true to allow entrance. Note that in the fail-safe functions that could kick the system out of that same ARMED state, those same parameters will effectively be OR-ed together in the logic. This is because only one failed guard function trigger parameter is necessary to fail out of that state. Thus, in this case there will effectively be seven transition functions any one of which will move the system out of the armed state. Since these pertain to entry into/or out of a safety-significant state, then these will need to be considered safety-significant functions.

Step 4. Identify Safety States and Modes. Continuing the analysis, the system engineer next will determine which states in the element state and mode transition diagrams are safety-significant states. Safety-significant states or Red states, as they will be termed in this document, are those states in which safety-significant functions can be performed. If all of the system functions that can be performed while in a particular state are not linked to a hazard in the FHA, then that is not a Red state. Red states are thus element states in which at least one state-bound function can be performed that is associated with at least one hazard. State bound means that that function will not be operative unless certain conditions are verified first. An example of a safety-significant function that is not state bound is something like obtaining environmental data (location, orientation, temperature, etc.) from sensors or BIT results or failure data from a device. Such data will be obtained regardless of state. Once the Red states are identified by examining the mapping of states to safety-significant functions in the FHA, the analyst updates the transition diagrams to show the safety states in another color such as Red. When we analyze the various element state diagrams for determination of safety-significant (Red) states, some elements will not have to do with the release of energy and have no hazards associated with them. An example of this is in the case of a missile system with an IR seeker, that element may have some state behavior, but will likely not have any Red states since no functions of the passive IR Seeker are associated with a hazard. Thus, no further safety analysis of such elements need be done.

Step 5. List Guard Functions. Guard functions are state transition functions (arrows on state diagrams) **that lead into** a Red state from an equal or less safety-significant state. These can be denoted with different colored arrows. For each element state machine that contains any Red states, the analyst will create a table of all guard functions. All the transitions into a safety-significant state must necessarily be safety-significant functions. This is because entry from a non-safety-significant (safe) state into a state where safety-significant functions can be enacted (e.g., a less safe state) is a safety-significant function since it brings the system functionally closer to being able to cause a hazard. These guards are safety-significant functions because failure of barring entry into a less safe state results in moving the system closer to a hazard.

Step 6. List Fail-safe Functions. Fail-safe functions are state transition functions (arrows on state diagrams) that lead **out of each Red state** and into a less safety-significant state and which are not part of the normal functioning of the system. More correctly, such transitions should be called fail-safer functions or fail-to-a-less-hazardous-state functions. It is important to regard all functions that act to move the system to less hazardous states as vital to a good safety design, even if these "fail-safes" do not take the system into hazard-free states. In the performance of the MSTHA, such transition functions will be termed fail-safes for utility. The fail-safes can be shown as differently colored arrows in element state diagrams. Similar to Step 5, in this step, for each element state machine that contains any Red states, we collect into a table all fail-safe functions. These transitions are the arrows on element state transition diagrams that lead from Red States to non-Red states or less safety-significant Red states. Note that if a trigger condition associated with a transition on a state diagram contains an "OR" in it, then there are actually two triggers since either terms in the "OR" statement can trigger the transition. These must be captured as two separate transition functions. Note that transitions out of one Red state into another equally or more safety-significant Red state is not a fail-safe. Note also that such systems as radiating RF Sensors, transitioning from some high-power radar mode to a lower power mode is not a fail-safe because it is part of the normal operation of the system and is in response to a command and not a failure. All of the fail-safe transitions are safety-significant functions, since failure of these functions may result in the system not going safe(r) when it should. That is, failure of a fail-safe is a hazard causal factor because it allows an unsafe system to remain in that unsafe state. Note that one way to determine if a particular transition state is acting as a fail-safe or just as is normal operation, ask the question "would a hazard result if the transition function failed to move out of the Red state in response to the trigger."

Step 7. Ensuring the FHA captures all Safety-Significant Transition Functions. At this point in the analysis, the analyst will have tables that contain all the existing safety-significant transition functions into and out of Red states along with the various element state diagrams with arrows depicting the state transitions. Step 7 essentially is a comparison of the transition functions in the FHA with those in the diagrams. Any discrepancies between the two will need to be explained since missing FHA functions associated with safety-significant state transitions are a safety finding. During this step, the SE will attempt to map all of the guard and fail-safe transition functions identified in Steps 5 and 6 with those FHA functions that were flagged in Step 3. The tables that contain the guard transitions and the fail-safe transitions will both include a column titled "Mapped to FHA" which is used to indicate the function ID number of the corresponding FHA function of the transition function that traces to each guard or fail-safe. If the analyst cannot map a particular guard or fail-safe transition function to one in the FHA, then "NO" is noted in the "Mapped to FHA" column indicating that there is no corresponding function in the FHA. Identification of guards and fail-safes that appear in the requirements, design and/or code but which do not appear in the FHA is an important safety finding, because this indicates that the design indicates transitions that were not captured in the FHA. This is particularly important since these particular missing functions are safety significant. If the code implementing these functions are not traced to any other hazard, then there is potentially a missing tagging of code as safety significant. Further, this may indicate missing LOR task completion. Also, if these functions are not included in the FHA, then perhaps the implementation of the safety-significant guard and fail-safe functions are missing from the code. This is a significant finding as well and should be captured as a safety-significant software defect and perhaps a missing safety-significant requirement. These are one of the first results of this type of analysis.

Extending System Understanding To Discovery of Missing SSFs
Next, the analyst attempts to identify missing guard and fail-safe functions. The upcoming Steps 8 and 9 are perhaps the most difficult of the fourteen steps in the Modes, States and Transitions Hazard Analysis because it is not a simple examination of what exists in the design or requirements but a search of what should be present but is missing. These two steps thus ask the analyst to use their knowledge of the system to extrapolate what other entry and exit triggers might be added to the system which are functionally similar to the existing guards and fail-safes already present in the system. During the analysis of what guards and fail-safes may be missing, a good method is to examine the existing ones and then deriving the reason(s) behind them.

When collecting the tables of Guard and Fail-safe transition functions, a column should be employed called “Reason for Transition.” When the rationale behind a fail-safe is described in the “Reason” field, the analyst can use that information to try to come up with other system parameters that may also satisfy that reason. A simple example is the condition of an aircraft being on the deck of the carrier. For an aircraft-launched missile system, ON DECK will be a trigger for preventing wing deployment and radar radiating. The reason is that when on deck personnel may be in the vicinity of the landed aircraft and so the kinetic energy of wing movement and the emitting of EM energy should be prevented.

Recall that a desirable system design is one that employs multiple sources for safety-significant system parameters used as safety-significant triggers. This is because at certain times in a real-world system one or more sources of this vital data may be unavailable or providing bad data, so redundancy is in keeping with designing a system for fault tolerance – a requirement for LOR-1 and LOR-2 software (ref. b). When a SE does this part of the analysis, they will be learning something about the system - learning what is it that really is preventing entry into a Red state or causing us to exit a Red state. The analyst will be gaining a deeper understanding of what is ultimately making those transitions safety critical. Whether the analyst wishes to go the extra mile or not, this type of analysis is useful to elucidate upon for the benefit of safety designers who wish to use this technique to design a safer system from the beginning.

This type of analysis hinges on a thorough understanding of the system and its component elements. To fully perform this step, the safety engineer will especially need to interface with system and Design Agent SMEs. In the discourse, the SE s should ask such questions as "What conditions and/or parameter values should make the weapon system disarm?" "Why does the system disarm if the air vehicle is too low or too high?" "Are there more than one source of vehicle altitude?" "Is every source of altitude used to trigger the system to disarm, or only the GPS value?" One of the hallmarks of the Modes, States and Transitions Hazard Analysis is the gleaning of system understanding by going through the state machines, identifying Red states, and identifying all safety transition functions so that we can make sure that an assessment has been done that all parameters that preclude entry into a Red state or exit from a Red state have been identified and if possible incorporated into the design. Once all of the safety parameters associated with entering or exiting Red states are gathered, the analyst determines whether they are in fact being employed as triggers. It may be that the system under analysis has been designed so that the obvious parameters are already being taken into consideration and used as transition variables (triggers) to prevent arming. However, we want to make sure that every parameter that has a value that precludes arming is within the set of transition variables. If not, then this fact should be captured as a finding (incomplete implementation of fault tolerant design).

Step 8. Determine if Sufficient Guards are in place. To perform this step, the analyst looks at each state and from an understanding of what the state means, conditions which are inconsistent with entering the state may be identified. If any new conditions are uncovered that should bar entry to a Red state based upon the value of existing system parameters or the addition of new parameters that can be obtained from the data already present in the system, then these new transition functions should be captured as a potential missing guard.

Step 9. Determine if Sufficient Fail-safes are in place. This step is essentially similar to Step 8 except that it pertains to the search for missing fail-safe functions. This step of the analysis looks at each Red state and from an understanding of what the state means, conditions which are inconsistent with staying within the state may be identified.

Step 10. Determine if Sufficient Aborts are in place. Best practices suggests that a system incorporates a method for the system to retreat from all unsafe states to a safe state either automatically or via operator action when the system detects a safety failure (ref. c, d). For this step, the analyst checks that all needed transitions to an aborted state exists for all unsafe (Red) states. Safety-significant states that are associated with particularly severe hazards should be designed with at least one automatic and/or operator-triggered abort. If such abort transitions could realistically and practically exist in the system but aren't part of the design, that is a finding of insufficient fault tolerance. Note that when aborted, a safety system isn't simply returned to another operational state such as the INIT state, since there are possible transitions out of that state. Abort states are different from other states because once entered they cannot be transitioned out of. This is to ensure that the element is not attempted to be used while in the aborted state. Abort states are thus terminal states. If abort states are implemented in the design, they should not be able to be transitioned out of and ones that can be moved out of are not correctly implemented Aborted states.

Interactions between Element States and Employment Modes

Step 11. Analyze Interactions between State Machines. In most complex systems there are usually more than one element with its own state machine and there are usually Employment Modes and/or Mission Modes each with a transition diagram or expected order of operations. The system can thus be in one Employment Mode and be in a certain Mission Mode and each element will be in one of its own internal states all at the same time. For systems with hazards, the safety engineer will wish to ensure that elements with Red states are fully cognizant of dangerous interactions between elements and modes and that the system correctly implements safety transition logic in the cases where elements clash. To accomplish this a special Table 1 is set up that is a matrix worksheet that facilitates the analysis of the Red states of the various safety-significant elements with all other element states and in our example with the various Employment Modes (Operate, Maintenance, Simulation).

Column one “State/Mode A” and column two “State/Mode B” in the Interaction Matrix list all the Red states in element A and then those of element B and so on along with the element name separated by a colon. In column 1 and 2 duplicates are omitted so that a matrix is formed to pairwise compare each state/mode with every other one. Note that column 1 includes only Red states while column 2 includes all element states. Column 3 “Safety Conflict between A and B” is a field that describes what possible conflict could occur between the two states/modes regarding safety functioning. In other words, column 3 describes some element A state functioning which should be prevented if element B is in the state shown in Column 2. Column 4 “Mitigating Function(s)” describes functions that would have the effect of preventing or lessening the severity of the safety conflict described in Column 3. Column 5 “Associated FHA Function ID” indicates whether the mitigating function(s) listed in Column 4 traces to an existing function in the FHA. Column 6 “Safety Tagged in FHA (Y/N)” indicates whether the FHA function(s) that trace to the mitigation in Column 4 is tagged as safety, i.e., traces to a hazard. Column 7 “Hazard Associated With Conflict” is the hazard associated with a failure to prevent or mitigate the conflict in column 3. If no hazard exists in the HTS for the conflict, “None” should be entered into Column 7. When filling out Table 1, especially Column 3, the safety engineer will look at what is meant by each Employment Mode and Mission Mode and encourages them to try to come up with scenarios where being in one state/mode or another might prevent the element from being in a particular state - one state/mode at a time.

Table 1 – Safety-Significant System Behavior Interaction Matrix

Red State A	State/Mode B	Conflict Between A and B	Mitigating Function(s)	Associated FHA Function ID	Safety Tagged in FHA (Y/N)	Hazard Associated With Conflict

After Table 1 is completed, there will be a number of potential findings. One finding is the identification of particular element states or Modes that should preclude certain other Red element states, but which are not implemented in the system as a state transition rule. Another finding is the determination that such rules may be implemented (as evidenced by a trace to the FHA) but which are not tagged as safety significant. System functions that prevent being in a particular Red element state while in a given other element state or mode are safety-significant functions. This step adds rigor to the association of safety-significant actions that should be taken when elements need to change state based upon changes in the system's Employment Mode or the element state of another system element. By examining the conflicts identified in Table 1 we can note that while there may be conflicts between a red state and another state such as in the first row, the conflict may not trace to a hazard. Rows two and three show that the state design accounted correctly for the conflict both in the state diagram and in the requirements which were presumably reflected as functions within the FHA. One of the findings found through this step is the identification of a conflict between states that while the state diagram design accounted for the conflict between the two states, this was not successfully translated into requirements and thus does not appear in the FHA. This often occurs when aspects of the design are emended after the FHA has been created. It may be that the mitigation exists in the requirements or the code, but not in the FHA. The SE will need to investigate what remedy is needed in these cases.

Step 12. Identify Potential New Hazards Due to Missing SSFs. As stated earlier, missing safety-significant functionality means that the system may be allowed to continue operating in an unsafe manner and potentially lead to the occurrence of a mishap. In this step, the SE determines whether a hazard associated with the missing SSFs already exists in the Hazard Tracking System (HTS). If no hazard exists related to the missing functions found in the above steps, new hazards must be created and entered into the HTS and processed like any other hazard. The analyst will collect the new hazards identified in this way indicating a description and a preliminary severity. This step then has the analyst go revisit any missing guard functions found in Step 8, any fail-safes found in Step 9, any missing abort transitions from Step 10 and any missing mitigations to state-state or state-mode interactions found in Step 11. The SE will thus attempt to associate the failure of each of them with an existing hazard in the HTS. If no hazard exists, then a new hazard needs to be suggested and captured for export as a finding. For example, in the case of an air-launched missile, if there was a missing transition from AUR state to MISSION ABORT as a result of the system detecting a Mission Processor Overheat failure, the hazard associated might be listed as “Launch of Unsafe missile due to failure to abort upon Mission Processor overheat.” Any new hazards that come from missing guard functions (Step 5) will stem from the system allowing entry into more hazardous states due to the safety-significant parameter. For example, for a RF element, a failure of the ON DECK guard between the sensor’s non-radiating STANDBY state and the ACTIVE RADAR HIGH PWR state could be captured as a hazard of “Personal exposure to harmful radar energy due to RF sensor radiating while On Deck.” Similarly, hazards should be drawn up related to the system remaining in Red states due to failure or lack of fail-safe system checks. Finally, the analyst should investigate hazards associated with missing logic that might prevent safety conflicts/interactions as found in Step 11.

Step 13. Ensure Safety-Significant Data used as Triggers is properly handled. Safety-significant data used as triggers for safety transitions such as guards and fail-safes and Aborts must be handled properly. Software functions that create, read in, modify and in any way affect safety-significant data are themselves safety-significant to the same degree. The analyst will start by creating a table listing all the data parameters involved with the transition functions flagged in the FHA. The data parameters used as triggers in safety-significant transition functions will necessarily be safety-critical data. Each **unique** data parameter will be listed in the table along with an indication of whether the data manipulation functions related to them are marked as safety-significant within the FHA. In other words, verify that the functions within the FHA that input and, in any way, modify the data are traced to the correct hazard. Let us say for example that we have identified that the parameter Height Above Target (HAT) is a factor in transitioning out of the Red 'Weapon Armed' state when less than 100 feet. Examination of the FHA may not have associated as safety-significant the functions that read altitude from an INS message and the function that queries Digital Terrain Elevation Data (DTED) from a table look up, or the function that calculates HAT in feet from these values. If these three functions were not traced in the FHA to the same hazard as the safety-significant function that transitions out of 'Weapon Armed' when the vehicle is too low, then there could be incomplete LOR task accomplishment associated with the software that implements those functions. The SE will then indicate in the table also any safety data that is not linked to a safety significant function. Secondarily, each piece of safety-significant data in a system should have at least one function which **checks or verifies** or at least sanity checks the data value before it is used in a safety-significant function. If any method could be practically implemented by the system to check each piece of safety data - especially safety-critical data as in LOR-1 and LOR-2 required tasks, then that check should be part of the system design. Sometimes there is no way to check a piece of data beyond sanity checking, type validity and boundary checks. Some techniques to verify safety-significant data can be accomplished by comparing parameters with multiple sources such as vehicle or missile altitude from GPS and from an altimeter. Calculations can also be done to derive values for comparison with the data requiring verification. For instance, heading or body vector can be checked by dead reckoning between the past several values and parameters such as speed can be checked against an effective speed computed from several past positions and the delta time elapsed. The computed check values may not be completely correct, but they can be used to verify the approximate value of the safety variable. The analyst should indicate in the table any data that is lacking a check that is not present in the design or requirements, but which may be technologically possible. Note also that functions that receive operator input data and button/control clicks in a Graphical User Interface (GUI) that affect safety functions are also safety-significant data-creation functions and thus will require checking functions.

Step 14. Assess Implementation of State Behavior. Note that the variable within a state-conscious safety system that indicates current state is a safety-significant parameter and must be verified prior to its use. A state machine within a software system can be implemented in a weak or a strong way. The weak implementation sets the current state to the initial state then checks the transitions into the next state and moves into the new state and then only sets the current state variable upon entry into the new state. Then the current state variable is checked before performing some state-bound functionality. LOR tasking guidance suggests that SwCI 1 or SwCI 2 software should be designed to be fault tolerant. Also, JSSSEH Appendix E (ref. c) and AOP-52 guidance (ref. d) states that safety should be checked periodically. This suggests that a strong implementation of the state machine should be employed for safety systems. A strong implementation is the same as weak except that in addition to checking the trigger parameters in order to go from the current state to the next state, a check should first be made to determine if the system is currently in the correct state. Also, a strong implementation may also require that all triggers into Red states be subjected to verification checks before implementing the transition function. Thus, a check is made of all conditions that lead to the current state and a check of conditions that trigger exiting the current state (if these are different). Guidance states that safety checks should be made periodically (ref. c, d). Checks of whether the system should transition out of a Red state should also be done upon the event when any trigger condition is detected. For instance, take the example that the system is in the 'Weapon Armed' and the system as implemented performs a verification check on the current state within a periodic function that runs at 1/2 Hz. One of the conditions that might cause an exit from the Weapon Armed' state might be orientation or location or vehicle speed which are obtained from the INS system at 25 Hz. It is better to transition out of the armed state upon the parameter change event – as soon as the INS sourced parameter determines it should, rather than waiting up to two seconds for the condition to be caught within the periodic check. This step in the analysis is an examination of how the state-behavior is mechanized in code and whether that implementation is sufficient. Results of this portion of the analysis are incomplete LOR tasking if sufficient fault tolerance is not in evidence. Another result is the identification of missing requirements associated with having a periodic check of the current state of the system when in a Red state.

Summary of Potential Findings

The analysis is concluded when all of the above steps are completed. Results out of this analysis are:

- Missing Safety Guard SSFs which are software/design defects injected at SRS stage (Step 8)
- Missing Fail-Safe SSFs which are software/design defects injected at SRS stage (Step 9)
- Missing abort functions from Red states that should have such functions (Step 10)
- Missing SSFs which prevent or mitigate dangerous interactions between the states of different elements and between states and modes. (Step 11). These should be tracked as safety-tagged software defects injected at the requirements phase.
- Identification of hazards (Step 12) associated with missing transition and abort functions (Steps 8, 9 and 10). These should be added to the Hazard Tracking System and processed as required by the standard software safety process.
- Missing SSFs pertaining to change, modification, use of safety data that are involved in Fail-Safe and Guard functions – this is safety data that is not linked to a SSF in the FHA (Step 13). This is a potential lack of LOR task accomplishment since these FHA functions may not be linked with hazards. These should be added to the FHA and processed as required by the standard software safety process.
- Functions in the FHA associated with safety data but not traced to hazards and thus not considered safety significant. (Step 13)

- Missing SSFs which are checks or verifications of safety-significant data (Step 13). These should be tracked as safety-tagged software defects injected at the requirements phase.
- Incomplete LOR task accomplishment associated with fault tolerance within state behavior implementation and missing periodic check of current state. (Step 14)

The documentation of the MSTHA will describe the findings along with a list of suggested follow-on actions that should be done, such as updates to the FHA, HTS and defect database for missing requirements/functionality and code LOR tagging. Because the safety engineer is not necessarily an expert on all aspects of the system, it is important during the first draft of the report of the MSTHA analysis to explicitly list any assumptions about the system that were made pursuant to the analysis so that these may be adjudicated and rectified so that follow-on analysis can understand all rationale used and perhaps redo steps that were done with incorrect assumptions.

In the performance of each of the above steps, no findings may be unearthed. This is a good result and is what may be expected when analyzing a well-designed system. Performing this analysis is thus like checking your math after a particularly complex division problem by multiplying. This analysis should, like the FHA, be an early design task to ensure that state behavior is explicitly created to bulwark safety within the system.

Conclusion

Fully assessing the safety of a system with complex behavior can be an especially difficult and elusive task. Viewing such systems as a set of states and state transitions is the best way to characterize their complex behavior. To accomplish this, safety engineers need an analysis technique that can organize the complexity into subcategories of behavior along logical lines. If a system with hazards employs complex variant behavior, it will be essential to perform some form of rigorous analysis that lays bare the detailed specifics of the state transition design and shows how that design is fault tolerant and employs all safety guidance and best practices. The methodology presented in this paper will accomplish this task. The above steps fastidiously followed will systematically generate the results needed to support the contention that a system's complex safety-significant behavior is understood, managed and the hazards associated with state and mode changes and interactions are fully analyzed and documented.