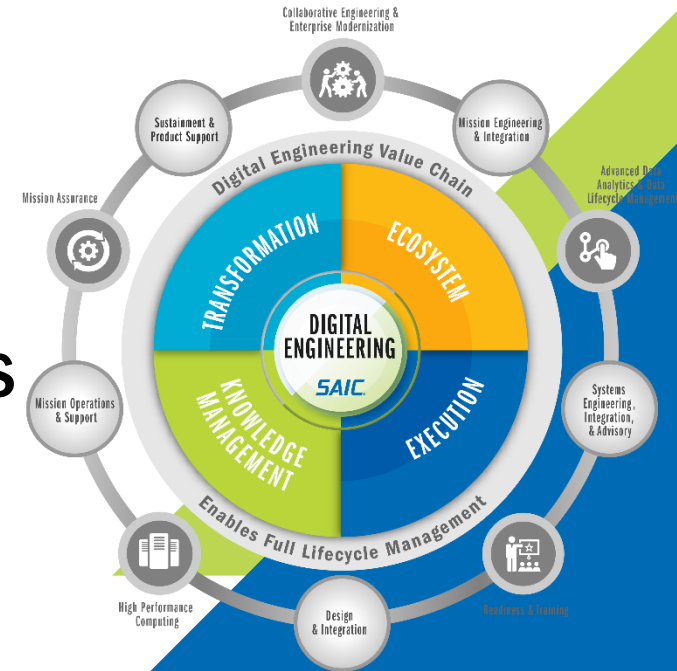


Using SysML State Machines to Automatically Conduct Failure Modes and Effects Analysis

2020 NDIA Systems & Mission Engineering Conference

Heidi Jugovic and Michael J. Vinarcik, P.E., FESD
Chief Systems Engineers
Solutions and Technology Group



This presentation consists of SAIC general capabilities information that does not contain controlled technical data as defined by the International Traffic in Arms (ITAR) Part 120.10 or Export Administration Regulations (EAR) Part 734.7-11.

SAIC[®]

FMEA/FMECA Basics

- FMEAs/FMECAs define failure modes and quantify the probability of their occurrence and the severity of their consequences.
- “Failures are prioritized according to how serious their consequences are, how frequently they occur, and how easily they can be detected. The purpose of the FMEA is to take actions to eliminate or reduce failures, starting with the highest-priority ones.” (American Society for Quality)
- This presentation will discuss:
 - Using model-based architectures as the authoritative source defining components and their functions
 - Defining failure modes and effects as sub-optimal states (SysML)
 - Leveraging tool capabilities to derive needed information
 - Providing data-driven analysis to replace/supplement mental calculations

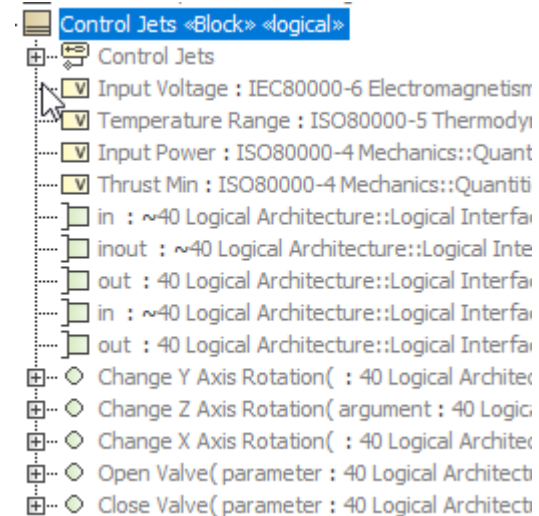
Historical Challenges

- Generating FMEAs/FMECAs may be viewed as a clerical, non-value added task by some engineers.
- ...this leads to “filing the serial numbers off” a previous FMEA/FMECA without rigorously reviewing it.
- Without rigorous review, the utility of the effort is greatly diminished.
- One aspect that increased the drudgery in document-intensive environments is that the engineer had to review multiple sources of information to ideate, qualify, and assess failure modes.
- Some FMEA/FMECA profiles have been released but they are not fully integrated into the system modeling process.

SAIC's Example Model: *Ranger* Lunar Probe

- Constructed in support of SAIC's Validation Tool
 - Rigorous, well-defined relationships enforced with automatic validation
 - Example structure: Functions are represented as operations owned by the performing block
- Original model is freely available for download:
<https://www.saic.com/digital-engineering-validation-tool>
- Modified for this topic to as proof of concept

All analysis that requires
components & functions should
reference this directly



Source Model Must be Well-Formed to Enable Analysis

Aspects of FMEA/FMECA Easily Addressed by Competent Modeling

- FMEA

- Define the system
- Define ground rules and assumptions to help drive the design
- Construct system Boundary Diagrams and Parameter Diagrams
- Identify failure modes
- Analyze failure effects
- Determine causes of the failure modes
- Feed results back into design process

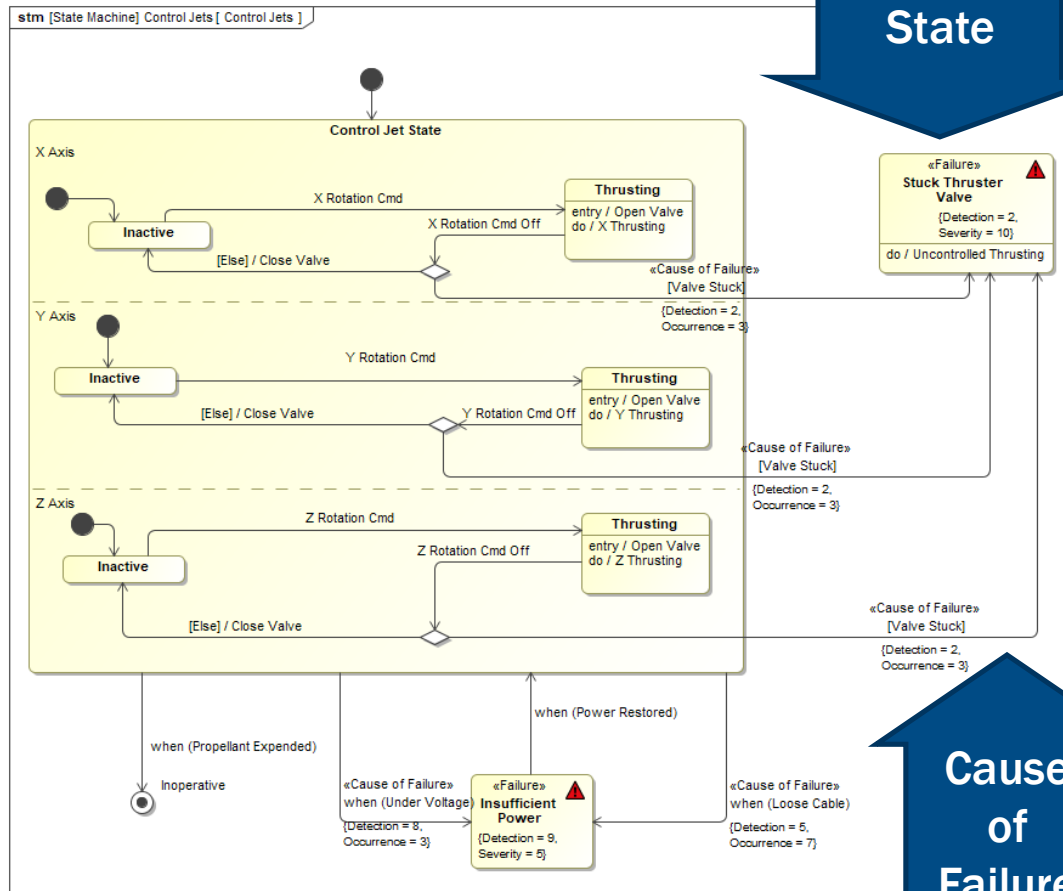
Aspects of FMEA/FMECA Easily Addressed by Competent Modeling

- FMECA Portion

- Transfer Information from the FMEA to the FMECA - becomes N/A
- Classify the failure effects by severity (change to FMECA severity)
- Perform criticality calculations
- Rank failure mode criticality and determine highest risk items
- Take mitigation actions and document the remaining risk with rationale
- Follow-up on corrective action implementation/effectiveness

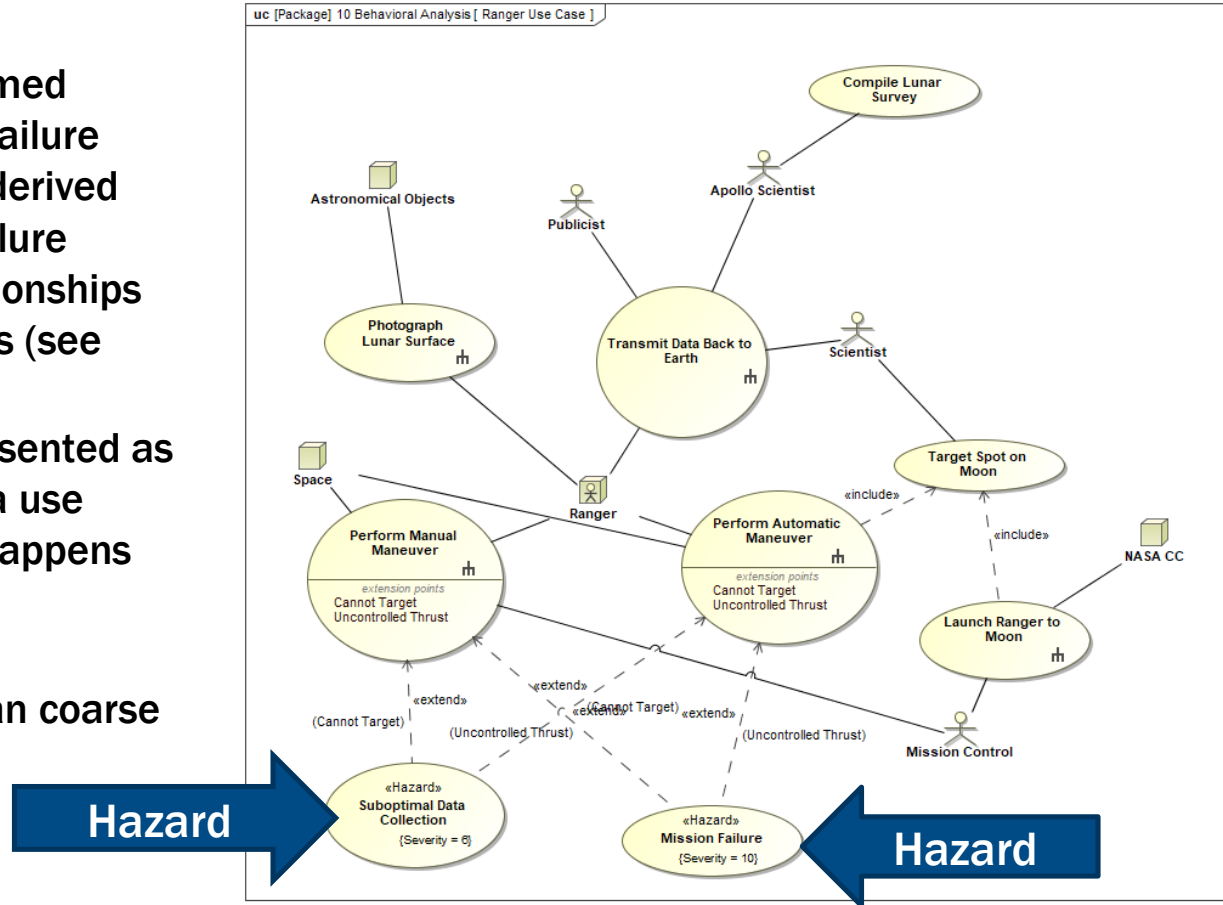
States and Failure Modes

- A failure mode is a sub-optimal state (state customized as **Failure**)
- Transitions that lead to a Failure are a **Cause of Failure**
- When a component enters a Failure state, some operations become unavailable, which are the local effect of failure
- When a component enters a failure state, additional undesirable behaviors may occur, which can be defined as operations customized as **Malfunctions**, and contribute to defining local effects of failure.



Use Cases, Hazards, and Final Effect of Failure

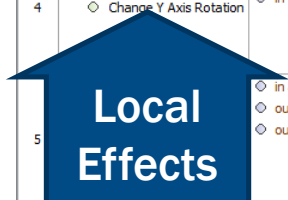
- When using a well-formed model, final effect of failure can be automatically derived from local effect of failure through inherent relationships to supported use cases (see next chart)
- A **Hazard** can be represented as an extension point of a use case, clarifying what happens when the use case is compromised
- A Hazard is assigned an coarse severity



Deriving Final Effect of Failure, Hazards, and Failure Severity

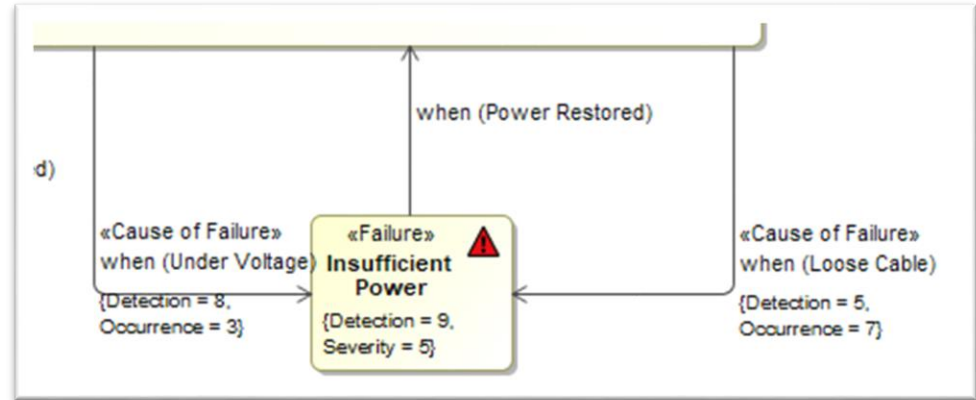
Each Failure state is assigned a severity by an engineer based reviewing the affected use cases, associated hazards, hazard severity, and owned Malfunctions.

#	Name	Owned Parameter	Available in State	Unavailable in State	Unavailable in Failures	Called On	Participates in (1st level up)	Participates in (2nd level up)	Participates in (3rd level up)	Possible Failures	△ Hazard	Hazard Severities
1	Close Valve	in parameter : Orientation Control Power		<input type="checkbox"/> Inactive <input type="checkbox"/> Thrusting <input type="checkbox"/> Control Jet State <input type="checkbox"/> Thrusting <input type="checkbox"/> Thrusting <input type="checkbox"/> Inactive <input type="checkbox"/> Inactive	⚠ Insufficient Power ⚠ Stuck Thruster Valve	<input type="checkbox"/> Close Valve <input type="checkbox"/> Close Valve <input type="checkbox"/> Close Valve						
2	Open Valve	in parameter : Orientation Control Power	<input type="checkbox"/> Thrusting <input type="checkbox"/> Thrusting <input type="checkbox"/> Thrusting	<input type="checkbox"/> Inactive <input type="checkbox"/> Control Jet State <input type="checkbox"/> Inactive <input type="checkbox"/> Inactive	⚠ Insufficient Power ⚠ Stuck Thruster Valve	<input type="checkbox"/> Open Valve <input type="checkbox"/> Open Valve <input type="checkbox"/> Open Valve						
3	Change X Axis Rotation	out : Control Thrust X Axis in : Orientation Propellant out result : Orientation Propellant	<input type="checkbox"/> Thrusting	<input type="checkbox"/> Inactive <input type="checkbox"/> Thrusting <input type="checkbox"/> Control Jet State <input type="checkbox"/> Thrusting <input type="checkbox"/> Inactive <input type="checkbox"/> Inactive	⚠ Insufficient Power ⚠ Stuck Thruster Valve	<input type="checkbox"/> Orientate Ranger <input type="checkbox"/> X Thrusting	<input type="checkbox"/> Maneuver	<input type="checkbox"/> Perform Automatic Mai <input type="checkbox"/> Perform Manual Maneu <input type="checkbox"/> Correct Course <input type="checkbox"/> Fire Main Engine	<input type="checkbox"/> Perform Automatic Maneuver <input type="checkbox"/> Perform Manual Maneuver	⚠ Cannot Target ⚠ Uncontrolled Thru ⚠ Cannot Target ⚠ Uncontrolled Thru	⚠ Suboptimal Data Collection ⚠ Mission Failure	6 10
4	Change Y Axis Rotation	out : Control Thrust Y Axis out : Orientation Propellant in : Orientation Propellant	<input type="checkbox"/> Thrusting	<input type="checkbox"/> Inactive <input type="checkbox"/> Control Jet State <input type="checkbox"/> Thrusting <input type="checkbox"/> Thrusting <input type="checkbox"/> Inactive <input type="checkbox"/> Inactive	⚠ Insufficient Power ⚠ Stuck Thruster Valve	<input type="checkbox"/> Orientate Ranger <input type="checkbox"/> Y Thrusting	<input type="checkbox"/> Maneuver	<input type="checkbox"/> Perform Automatic Mai <input type="checkbox"/> Perform Manual Maneu <input type="checkbox"/> Correct Course <input type="checkbox"/> Fire Main Engine	<input type="checkbox"/> Perform Automatic Maneuver <input type="checkbox"/> Perform Manual Maneuver	⚠ Cannot Target ⚠ Uncontrolled Thru ⚠ Cannot Target ⚠ Uncontrolled Thru	⚠ Suboptimal Data Collection ⚠ Mission Failure	6 10
5		in argument : Orientation Propellant out parameter : Orientation Propellant out parameter 1 : Control Thrust Z Axis	<input type="checkbox"/> Thrusting	<input type="checkbox"/> Inactive <input type="checkbox"/> Thrusting <input type="checkbox"/> Control Jet State <input type="checkbox"/> Thrusting <input type="checkbox"/> Inactive <input type="checkbox"/> Inactive	⚠ Insufficient Power ⚠ Stuck Thruster Valve	<input type="checkbox"/> Orientate Ranger <input type="checkbox"/> Z Thrusting	<input type="checkbox"/> Maneuver	<input type="checkbox"/> Perform Automatic Mai <input type="checkbox"/> Perform Manual Maneu <input type="checkbox"/> Correct Course <input type="checkbox"/> Fire Main Engine	<input type="checkbox"/> Perform Automatic Maneuver <input type="checkbox"/> Perform Manual Maneuver	⚠ Cannot Target ⚠ Uncontrolled Thru ⚠ Cannot Target ⚠ Uncontrolled Thru	⚠ Suboptimal Data Collection ⚠ Mission Failure	6 10



Assigning & Calculating Probabilities of Occurrence

- Each Cause of Failure is assigned a Occurrence and Detection
 - In this example, we use a scale of 1-10 from ASQ, but more precise numerical values can be applied (<https://asq.org/quality-resources/fmea>)
- Each Cause of Failure has a Severity which is pulled from the Severity of the Failure state (see previous)
- Each Cause of Failure has a calculated Criticality and RPN



Causes of Failure	Cause Detection	Cause Occurrence	Cause Criticality	Cause RPN
⚡ Trigger:when (Under Voltage)	8	3	24	120
⚡ Trigger:when (Loose Cable)	5	7	35	175

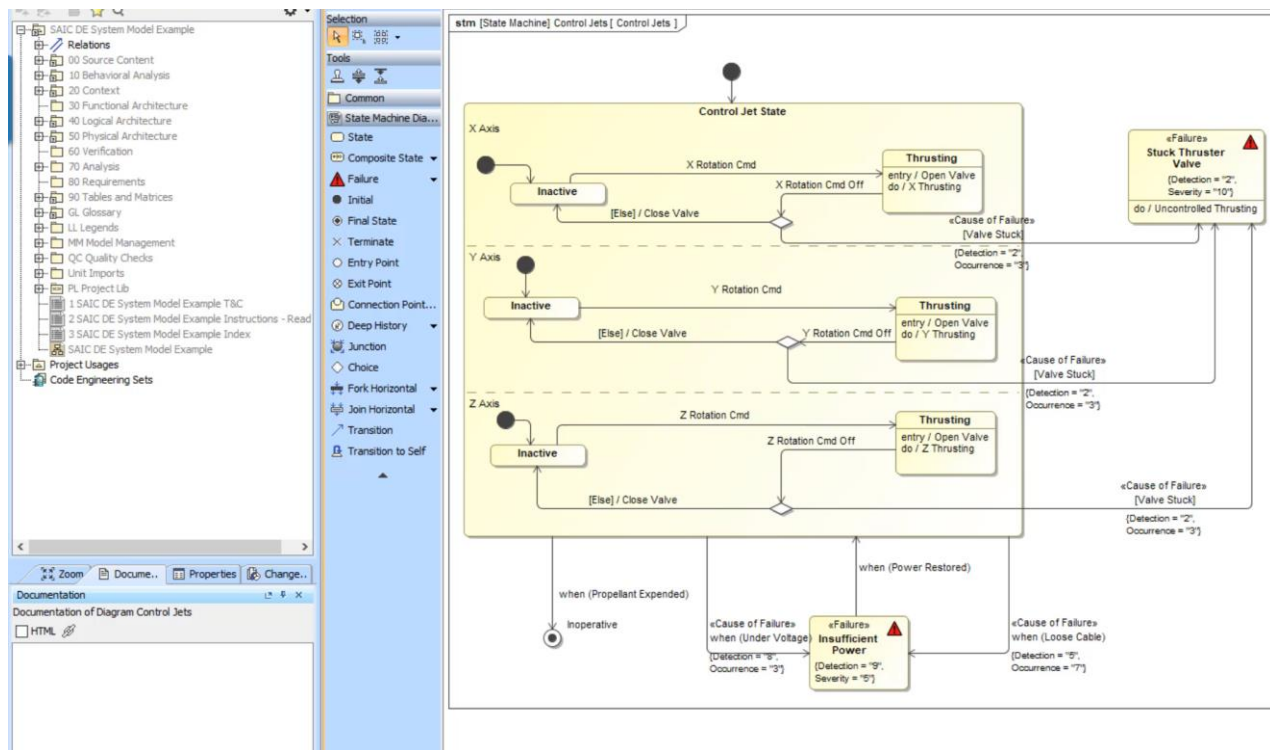
Finalizing the Analysis

All Cause of Failure transitions to a Failure state contribute the Criticality and RPN

#	Name	Severity	Maximum Occurrence	Maximum RPN	Causes of Failure	Cause Detection	Cause Occurrence	Cause Criticality	Cause RPN	Malfunctions	Unavailable Functions	Affected Use Cases	Failures	Hazard	Hazard Severity
1	⚠ Insufficient Power	5	7	175	⚠ Trigger:when (Under Voltage)	8	3	24	120		◇ Close Valve(parameter : C ◇ Open Valve(parameter : C ◇ Change X Axis Rotation(: ◇ Change Y Axis Rotation(: ◇ Change Z Axis Rotation(a ⚠ Uncontrolled Thrusting(:	◇ Perform Automatic Maneu ◇ Perform Manual Maneuver	◇ Cannot Target ◇ Uncontrolled Thrust ◇ Cannot Target ◇ Uncontrolled Thrust	◇ Suboptimal Data Collection ◇ Mission Failure	5 10
					⚠ Trigger:when (Loose Cable)	5	7	35	175						
2	⚠ Stuck Thruster Valve	10	3	60	{} Valve Stuck {} Valve Stuck {} Valve Stuck	2	3	6	60	⚠ Uncontrolled Thrusting(◇ Close Valve(parameter : C ◇ Open Valve(parameter : C ◇ Change X Axis Rotation(: ◇ Change Y Axis Rotation(: ◇ Change Z Axis Rotation(a	◇ Perform Automatic Maneu ◇ Perform Manual Maneuver	◇ Cannot Target ◇ Uncontrolled Thrust ◇ Cannot Target ◇ Uncontrolled Thrust	◇ Suboptimal Data Collection ◇ Mission Failure	5 10

- Additional calculations can roll up these values to any useful level
- Results can be displayed and manipulated in the model
- Changes to values propagate immediately
- Note the *malfunctions* that occur in the failure state (in this case, “Uncontrolled Thrusting”)

Demonstration



A Few Additional Considerations

- **Consider cross-discipline applicability:**
 - Traditional hardware failure/software types
 - Overlaps with Security failure modes. Data is:
 - Late: State timing triggers
 - Unavailable: consider which states are never entered if data is not received? What behavior is not available? Use the model to identify the interfaces involved.
 - Manipulated by an adversary
 - Intercepted and made available to an adversary
 - Overlaps with Safety failure modes

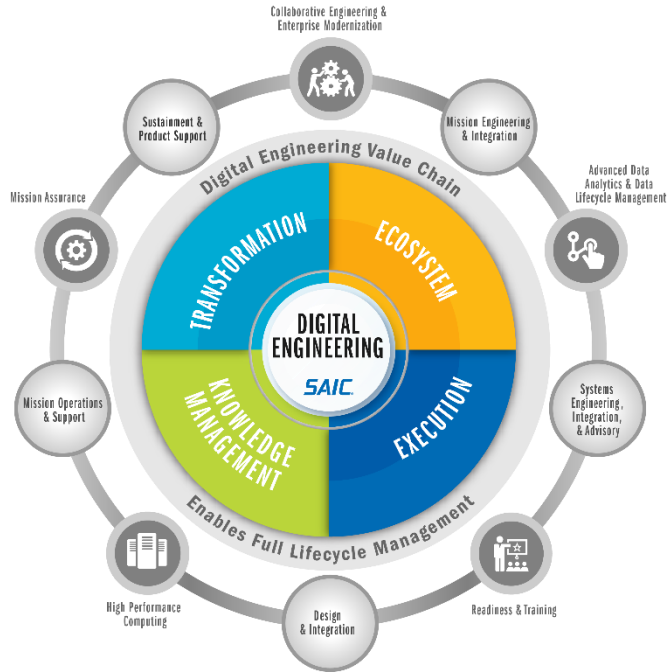
A Few Additional Considerations

- Normal transitions can represent recovery from a Failure state
- 0* transitions are allowed
- Any transition trigger is allowed (e.g., TimeEvent, SignalEvent, etc.)
- Interdiction of transitions can be used to document controls (see *Interdiction: The Application of SysML State Machines to Cybersecurity*, Vinarcik and Colwander, 2018 NDIA Systems Engineering Conference)

The Hidden Benefit

- By adopting this approach, FMEA/FMECA analysis is embedded into the architecture and uses the same information (so it is always synchronized).
- The hazard/FMEA/FMECA analysis can also be segregated into a higher-level analysis model that uses the primary architecture model:
 - Prevents proliferation of properties/domain-specific additions into the primary architectural model
 - Isolates potentially sensitive/trade secret information

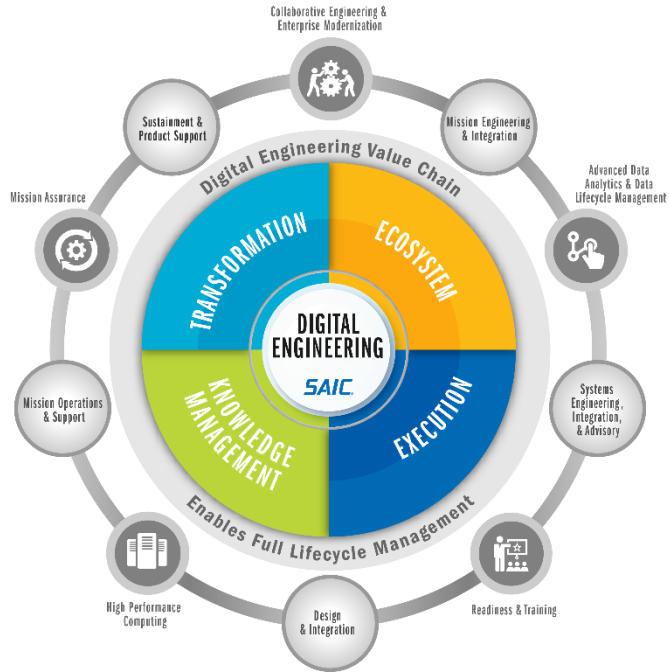
SAIC Digital Engineering



Online: saic.com/digital-engineering

Email: digitalengineering@saic.com

SAIC Digital Engineering



SAIC DE Profile & Validation Rules:

<https://www.saic.com/digital-engineering-validation-tool>