Improving Reliability, Safety and Mission Assurance using Early Visibility Metrics

Lucas Layman, Forrest Shull, Victor Basili

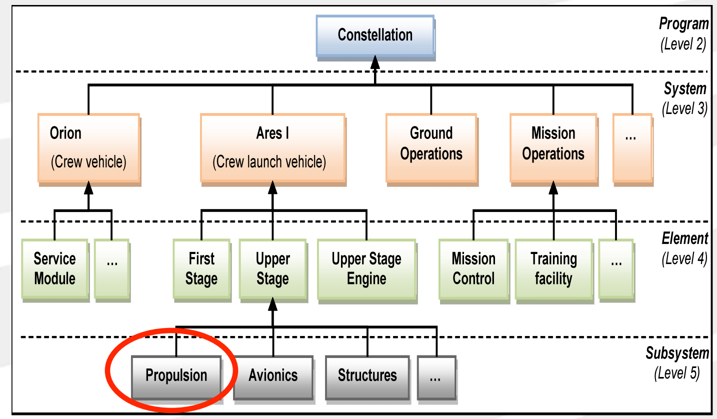
Fraunhofer Center for Experimental Software Engineering

Fraunhofer USA
Center for Experimental Software Engineering

Challenge

Where in my system is the greatest risk?

- *Where is the greatest security risk in my supply chain?*
- *Which subsystems are most prone to safety concerns?*
- *What is the reliability of my communication system?*



How do I quantify system and software risk when the system and software do not yet exist?

– We rely on our processes and experts to answer these questions during development

How can we gain early insight into reliability, safety and mission assurance risks in a more concrete manner?

Fraunhofer USA
Center for Experimental Software Engineering

Example: obtaining early insight into software safety on Constellation

The **Constellation program** is NASA's next generation human spaceflight program.

CONSTELLATION

Orion crew vehicle

Ares rockets

NASA objective: to quantify software safety risk in the Constellation program from a management perspective

- *Which systems and subsystems have the greatest software safety risk?*
- *How can we measure software safety risk?*
- *Are our processes appropriate for and being performed appropriately to achieve software safety?*
- We examined three spaceflight hardware systems during Phase A development

Managing risk during development

Reliability, Safety and Mission Assurance (RSMA) processes are the most common defense against system risks:

- Technical risk – flaws in the design and implementation that lead to system failure, loss of mission, or loss of life.
- Process risk – risks that emerge when:
  - The RSMA processes are not performed appropriately (we are **NOT** talking about being process police!)
  - The RSMA processes are not well-defined
  - The RSMA processes are not appropriate for the situation

Technical risk

- The Login system is highly susceptible to external attack
- The system uptime is predicted to be less than five 9's
- The flight computer has a single point of failure in the avionics control bus

Process risk

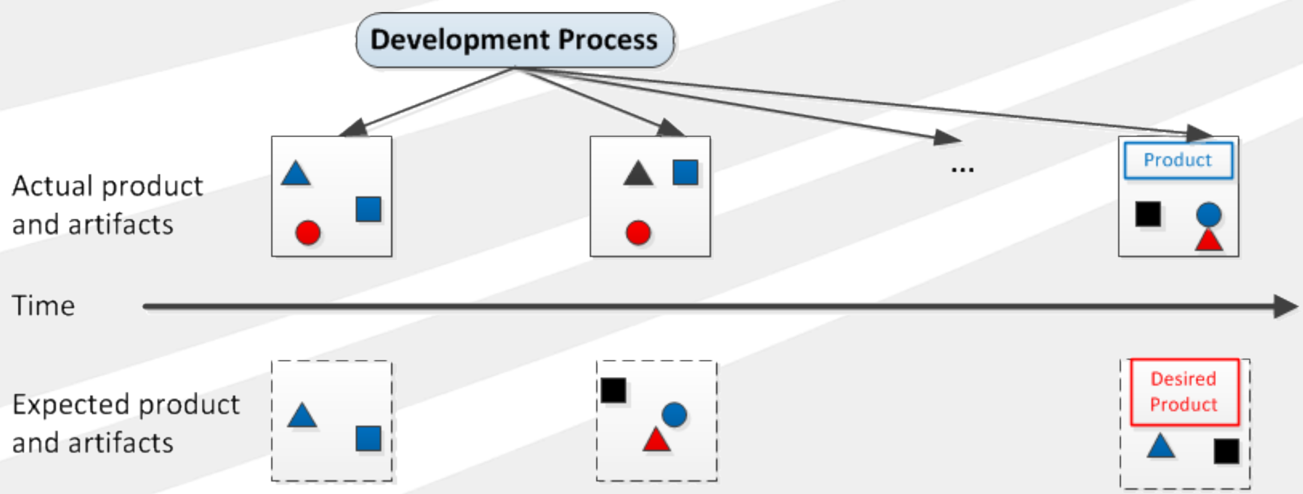
- Staff are not recording necessary information in attack graphs
- The reliability models do not apply to distributed systems
- The process for performing FMECA analysis on software is not clear

Fraunhofer USA
Center for Experimental Software Engineering

Risk Measurement Approach

Approach: Measure process artifacts with respect to the risks they are meant to mitigate.

- Process artifacts contain indicators of potential technical risk.
- Processes and process artifacts are available throughout development.
- Quantifiable measures for trend analysis, baselines and comparison



The Technical and Process Risk Measurement methodology

This method was developed to address software safety risks on the DoD's FCS and NASA's Constellation programs

Six step Technical and Process Risk Measurement (TPRM) methodology:

1. Identify **insight areas or intermediate artifacts**
2. Identify the **measurement opportunities**
3. Develop **readiness assessment questions**
4. Define **goals and questions** for each risk area
5. Develop and enumerate **measures and models** of how they will be **interpreted** via threshold values.
6. Propose **responses** to identified risks

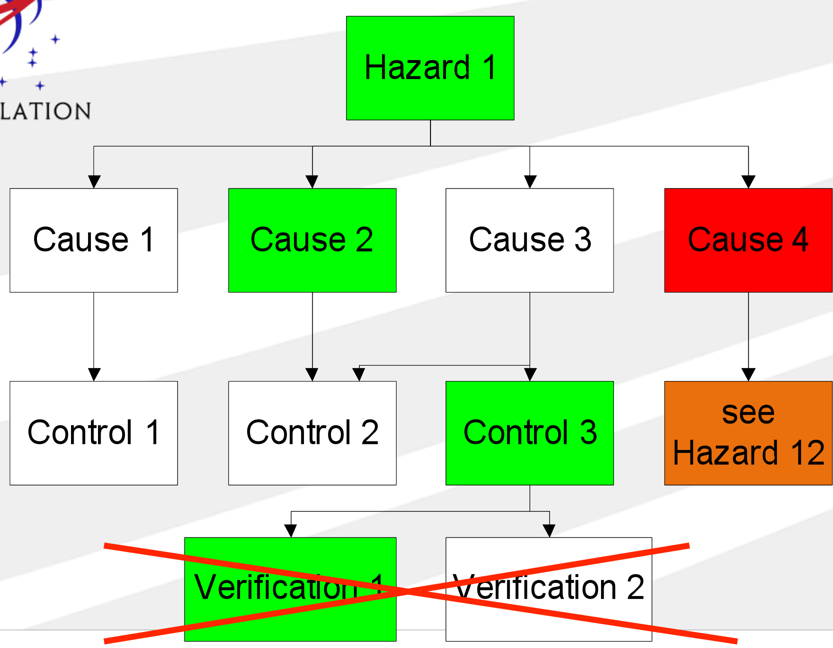
What can we measure?

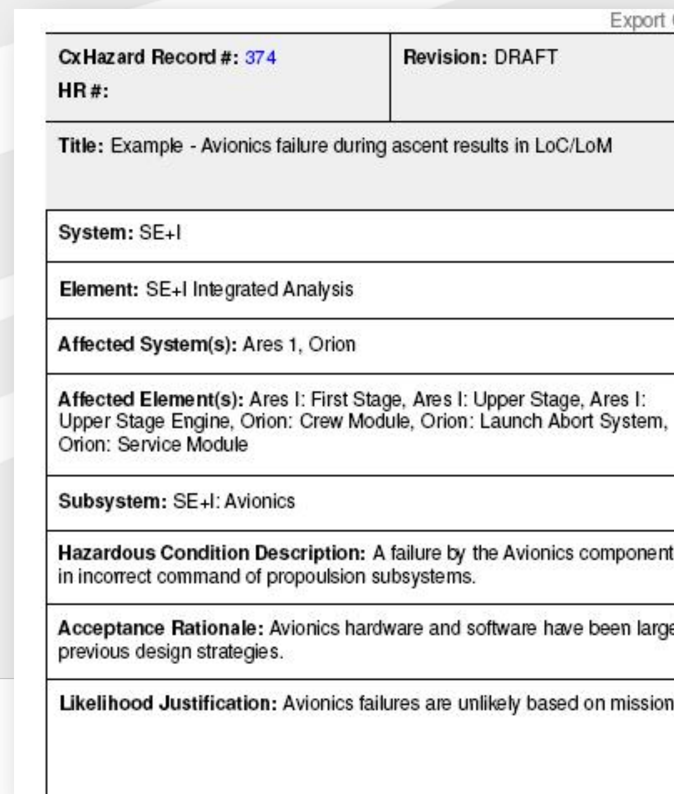
Step 1: Identify insight areas from the RSMA processes that provide insight into risks.

Step 2: Identify the measurement opportunities that provide insight into each risk area.

Step 3: Develop readiness assessment questions to provide a quick status of the risk and to identify if it is possible to delve deeper into the area?

CONSTELLATION





Export C

CxHazard Record #: 374 **HR #:**	**Revision:** DRAFT
Title: Example - Avionics failure during ascent results in LoC/LoM	
System: SE+I	
Element: SE+I Integrated Analysis	
Affected System(s): Ares 1, Orion	
Affected Element(s): Ares I: First Stage, Ares I: Upper Stage, Ares I: Upper Stage Engine, Orion: Crew Module, Orion: Launch Abort System, Orion: Service Module	
Subsystem: SE+I: Avionics	
Hazardous Condition Description: A failure by the Avionics components in incorrect command of propoulsion subsystems.	
Acceptance Rationale: Avionics hardware and software have been large previous design strategies.	
Likelihood Justification: Avionics failures are unlikely based on mission	

Fraunhofer USA
Center for Experimental Software Engineering

Defining risk measures

Step 4: Define goals and questions for each risk area to expose risks associated with RSMA process artifacts.

Step 5: Develop and enumerate measures and models of how the metrics will be interpreted via threshold values.

CONSTELLATION

Goal: C s.

– causes

–

Constellation highlights

- Examined 154 hazard reports, 2013 causes, 4096 controls
- ~60% of hazards are software related
- 7% of hazards have "hidden" software risk
- 30% of causes and 17% of controls were *transfers*

Hazard re					**related**
Affected					
Avionics					0%
Main Propulsion sys	34	12	18	35%	53%
Roll reaction control	29	9	14	31%	48%
Thrust vector control	15	5	5	33%	33%

Responses to identified risks

Step 6: Propose responses to identified risks.

Risks identified through measurement

- Lack of consistent scope in describing software functions impairs risk assessment.
- Incorrect references to hazard reports, causes and controls impair traceability
- Ubiquity of transferred causes and controls mask software risk
- …

Responses implemented by program

- Creation and dissemination of a "user guide" for specifying software causes.
- Issue "letters of interpretation" of hazard analysis process
- Additional training sessions for safety engineers
- Automated verification of references in the Hazard Tracking System
- HTS functionality to identify software causes and controls

Fraunhofer USA
Center for Experimental Software Engineering

Software cause "user guide"

Documenting software causes and controls in hazard reports

As part of an OSMA SARP Safety Metrics Initiative, NASA SR&QA personnel and the Fraunhofer Center for Experimental Engineering have put together the following guide to assist safety engineers in **documenting software causes and controls of hazards in hazard reports for Phase I safety reviews.**

Four attributes must be specified for *each software cause and sub-cause* documented in hazard reports:

Specificity

1. **Index the cause** – Label each software cause and sub-cause with a unique identifier.
2. **Identify the origin** – Indicate the CSCI that fails to perform its operation correctly.
3. **Specify the erratum** – Provide a description of the erroneous command, command sequence or failed operation of the CSCI.
4. **Specify the impact** – Provide a description of the erratum's effect that, if not controlled, results in the associated hazard. If known, identify the specific CSCI(s) or hardware subsystem(s) affected.

Attribute	Example acceptable values	Unacceptable values
Index	If cause 8 is "Software based control error," label sub-causes as 8.a, 8.b, 8.c, ...	*{Software sub-causes not indexed}*
Origin	– Avionics CSCI – Propulsion CSCI – Vehicle Management CSCI – Timeline Management CSCI	– the software – the Flight Computer – a computer based control error – a general software fault
Erratum	– failure to detect a problem	– the software fails

Fraunhofer USA
Center for Experimental Software Engineering

Main Contributions

TPRM methodology leverages process artifacts to gain early insight into insight into reliability, safety and mission assurance

Completed two case studies applying the TPRM methodology: Future Combat Systems and Constellation

- Identified four risks in the hazard analysis process for FCS; six risks in the Constellation process.

Created a baseline for comparison with future review milestones and future NASA projects

- Metrics provided to identify subsystems and mission phases with the greatest potential software safety risk

Next Steps

We can apply this approach to processes meant to achieve other "ilities":

– Reliability
– Security
– Mission assurance
– Costs
– …
– Any process with intermediate artifacts whose purpose is to achieve the desired characteristics

We are looking for collaborations with organizations, programs and projects with such processes in place.

Thanks and acknowledgement

Contact: Lucas Layman (llayman@fc-md.umd.edu)

FCS article:

http://www.cs.umd.edu/~basili/publications/journals/J112.pdf

Constellation technical report:

http://www.fc-md.umd.edu/TR/Safety-metrics_TR_10-101.pdf

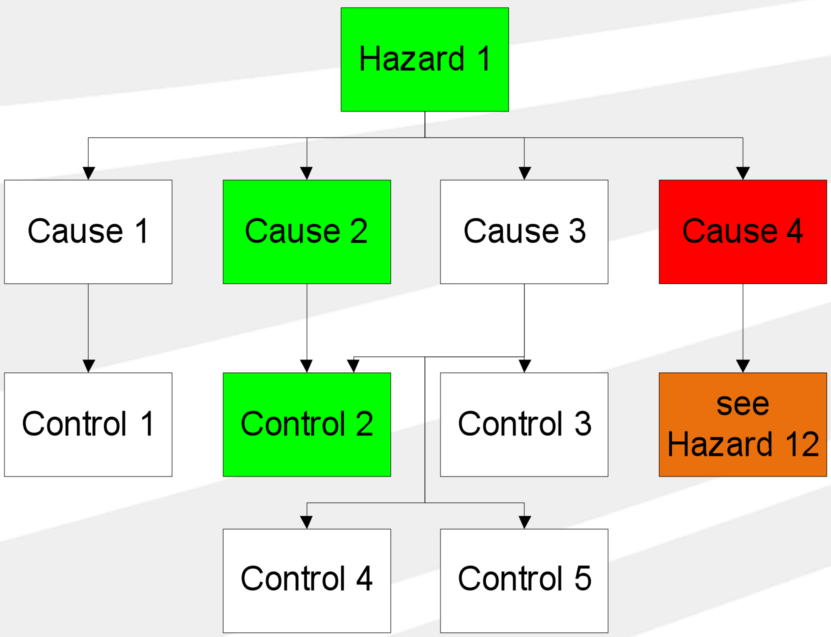
Acknowledgements

- This research was supported by NASA OSMA SARP grant NNX08AZ60G
- Thanks to Karen Fisher and Risha George at Goddard Space Flight Center
- Thanks to Frank Marotta at Aberdeen Proving Grounds

Analysis method

<u>Goal 1</u>: to quantify the relative importance of software with respect to system safety.

- **Software-related cause or control** describes software behavior
- **Software-related hazard** has one or more software causes or controls



Hazard Report	Cause	Controls					
		1	2	3	4	5	6
Hazard 1	1						
	2						
	3						
	4						

Software hazard, cause or control	
Non-software control	
Transferred cause	
Transferred control	

Number of software causes

Ares US

	Non-software cause		Software cause	
no software control	393	71%	0	0
at least 1 software control	76	14%	86	15%
Transferred causes	252			
Total	806			

Orion

	Non-software cause		Software cause	
no software control	402	77%	0	0%
at least 1 software control	57	11%	62	12%
Transferred causes	151			
Total	672			

J-2X

	Non-software cause		Software cause	
no software control	275	81%	0	0%
at least 1 software control	9	3%	57	17%
Transferred causes	194			
Total	535			

Center for Experimental Software Engineering

Number of software controls

Ares US

	N	% of total	% of non-transferred
Non-software	1603	64%	82%
Software	243	10%	12%
Generic software controls	105	4%	5%
Transferred controls	566	22%	-
Total	2517		

Orion

	N	% of total	% of non-transferred
Non-software	1802	75%	85%
Software	298	12%	14%
Generic software controls	37	2%	2%
Transferred controls	262	11%	-
Total	2399		

Fra

Center for Experimental Software Engineering

Goal 2: Level of Risk – Initial study

- **Goal 2**: Quantify the level of risk presented by software in the Constellation program.

Ares US

Hazard ratings

	#	%
La	5	18%
Lb	7	25%
Lc	7	25%
Ld	3	11%
Le	6	21%

Cause ratings

	#	%
L1	65	50%
L2	26	20%
L3	38	29%

Orion

	#	%
La	3	8%
Lb	1	3%
Lc	14	38%
Ld	13	35%
Le	6	16%

	#	%
L1	65	38%
L2	68	40%
L3	37	22%

Process risks and recommendations

- Inadequate thruster performance results in loss of control"
- Control 29 has 14 "sub-controls"
- "Human error" is actually Cause 15

Cause: CAUS11 — Software Based Control Errors

a. Pressure in the propellant tanks is controlled by Propulsion based on the inputs of redundant pressure transducers in each tank to maintain the proper quantity of propellant being delivered to the thrusters.

b. The above-mentioned functionality will be implemented as requirements in the Software Requirements Specs for Propulsion, System Management, and Displays and Controls, then flowed into design and code and undergoes a test and validation process. See [redacted] for details on process controls in the Orion software development process for: Requirements Defects, Design Defects, and Code Defects).

Cause 11: Human Error

In order to implement this functionality, the Controls software performs the following processing:

a. Rejects any inputs from [redacted] The above-mentioned functionality will be implemented as requirements in the Software Requirements Spec for Controls, then flowed