SOFTWARE FAILURE MODES EFFECTS ANALYSIS OVERVIEW

- Copyright, Ann Marie Neufelder, SoftRel, LLC, 2010
- amneufelder@softrel.com
- www.softrel.com

This presentation may not be copied in part or in whole without written permission from Ann Marie Neufelder

Table of Contents

Software Failure Modes Effects Analyses Defined

- Analysis is adapted from Mil-STD 1629A, 1984 and Mil-HDBK-338B, 1988
- Can be applied to firmware or high level software
- Software development and testing often focuses on the success scenarios while SFMEA focuses on what can go wrong
- More effective than traditional design and code reviews because
  - Reviews often focus on style instead of failure modes
  - Reviews often identify issues but not the system wide effects of the issues
  - Reviews are often not targeted to high risk areas

Software failure modes….

- Software failure modes are generally either
  - Data related
  - Event related
- Many of these are repeatable
- Many of these cannot be corrected once the failure event is encountered
  - So hardware redundancy is often not a corrective action
  - Failure modes that might be corrected or avoided with hardware redundancy are indicated with an “&” in class

Software FMEAs can be conducted from 6 different viewpoints

FMEA viewpoint	Product Level Viewpoint	Identifies failures related to.	Life cycle timing
Functional	Requirements	Timing, sequence, Faulty data, erroneous error messages for a component	SRS completion
Interface	Interface between 2 components	Timing, sequence, Faulty data, erroneous error messages between 2 components	Interface Design Spec completion
Detailed	At class or module level	All of the above plus memory management, algorithms, I/O, DB issues	Detailed design or code is complete.
Production	Process related failures during development	Problems with many defects and/or ability to meet a schedule, execution and tools	Any time
Maintenance	Changes to the software	Problems when software is modified , installed, updated	During maintenance
Usage	User friendliness and consistency, documentation	Software/documentation is too difficult or inconsistent to be used properly	As early as possible as these issues will influence design

The cost of doing a SFMEA

- What are the technical benefits?
- Who will do the SFMEA?
- How much time will it take?
- Are the benefits worth the cost?
- Common SFMEA mistakes that can cost money

When properly implemented at the right point in the lifecycle Software FMEAs can…

- Make requirements, design and code reviews more effective
- Identify single point failures due to software
- Identify defects that cannot be addressed by redundancy or other hardware controls
- Identify abnormal behavior that might be missing from the requirements or design specifications
- Identify unwritten assumptions
- Identify features that need fault handling design
- Address one failure mode could mean eliminating several failures

What personnel is required for a SFMEA?

Personnel	Strengths
Facilitator	Understands the SFMEA process
Software management	Responsible for the software project
Software engineers	Key engineers with subject matter expertise for the product being analyzed. Depends on viewpoint: •Functional SFMEA- someone who is familiar with the SRS is required. •Interface SFMEA -the person(s) who designed the interfaces. •Detailed SFMEA -the person responsible for design and coding.
Domain experts	These are people who are knowledgeable of how the system will be used and what kinds of events are most critical to an end user or system

What is the typical effort required for each part of the SFMEA?

Task	Functional, interface or detailed SFMEA	Personnel involved with this task
Planning	Can usually be done in a half day	All
Collect actual software failure data to identify likely failure modes	Usually 1 day	Facilitator
Construct left side of SFMEA table	Depends on viewpoint •Functional - 30-60 mins for each SRS statement •Interface - 30-90 mins for each interface variable •Detailed - 30-90 mins for each module	Facilitator does initial work. Software engineers review for completeness.
Effects on system, likelihood, severity	Can take up to 15 minutes per failure mode	All – Facilitator keeps discussion moving
Mitigate risks/make corrective action	Entirely dependent on the corrective action	Software management

These are some of the benefits that my customers have experienced from the SFMEA analysis

The SFMEA is particularly cost effective at finding a small number of defects that have catastrophic consequences and/or will result in many failures by many end users

- Project X – Safety/monetarily critical equipment - A small number of very serious defects were uncovered that would have been difficult if not impossible to find in testing. The cost of these defects being discovered even once in the field would have been several million. The cost of the analysis was 28K.
- Project Y – A web based system allowed non-paying customers to sometimes (under certain conditions) retrieve a product without paying first. The testing had been directed to the positive case (paying customers get their product) and not the negative case. That's because the SRS never stated what the system should "Not" do. This defect would have resulted in significant loss of revenue if deployed.

Common SFMEA mistakes that cost money and reduce benefit

- Starting at the wrong place
  - Usually you do not *start* the analysis at individual lines of code
- Doing the analysis too late in the life cycle
- Assuming that certain failure modes won't happen before analyzing them
- Neglecting to tailor the list of failure modes to your application type
- Neglecting to filter/rank the code by risk and impact
- Assuming that hardware redundancy will prevent all software failure modes
- Neglecting to decide on the best viewpoint before doing the analysis

What is the typical effort required for the entire SFMEA?

- A typical project has the below SFMEA expenditures
- Things that make SFMEA analysis go faster and better
  - More detailed product documentation such as SRS, IDS, design docs, etc
  - Software engineers who are willing to think about how the software can fail instead of trying to prove that it can't

Personnel	Strengths
Facilitator	150-200 hours
Software management	20-30 hours not including time required to correct issues
Software engineers	36-60 hours not including time required to correct issues
Domain experts	20-40 hours

About Ann Marie Neufelder, SoftRel, LLC

- Has been in software engineering since 1983
- Authored the NASA webinar on software FMEA
- Has been doing SFMEA for 25+ years
- Has completed software/firmware FMEAs in these industries and applications
  - Commercial and defense vehicles
  - Drilling equipment
  - Electronic warfare
  - Ground based satellite systems
  - Lighting systems
  - Commercial appliances/electronics
  - Space systems

What you will get from the 1 day SFMEA class

- Hands on step by step process for doing the SFMEA within schedule and cost constraints
- Templates to facilitate
  - Completion of each step of the SFMEA process
  - Brainstorming process (the most difficult step)
  - 300 failure mode/root cause pairs to pick from
- Examples of completed SFMEAs from real world
  - Seeing a real example in various stages of construction is the most valuable step towards constructing a *useful* software FMEA
  - Examples of how NOT to do a SFMEA

Software FMEA services provided by Ann Marie Neufelder

- The hardest part of the SFMEA is getting it started
- The second hardest part is knowing how to keep it under budget
- Ann Marie Neufelder can help with that
  - Facilitating an effective SFMEA to ensure minimum $ spent
  - Performing a RCA to identify most likely failure modes/root causes
  - Laying out the left side of the SFMEA
  - Working with the software engineers and domain experts to complete right side of SFMEA
  - Keeping the analysis on schedule by leading the discussions down the most productive path

What the other consultants don't have

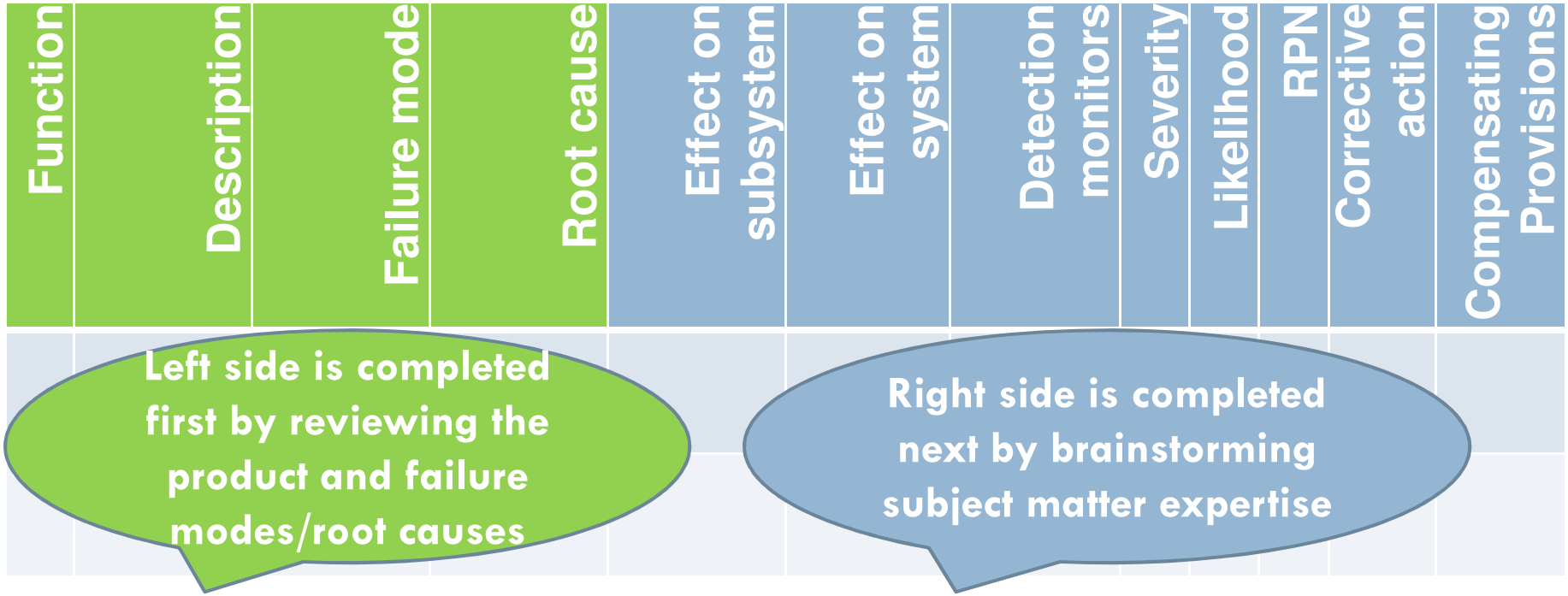
- Ann Marie Neufelder has identified more than 300 failure mode/root cause pairs
- She has applied SFMEAs on real world software for all 6 viewpoints
- Since she has 25+ years of software engineering experience she knows how to integrate this analysis on real world versus academic software projects
- Since she has 25+ years of experience, she knows the common mistakes that will kill the effectiveness of a SFMEA

Technical aspects of the SFMEA

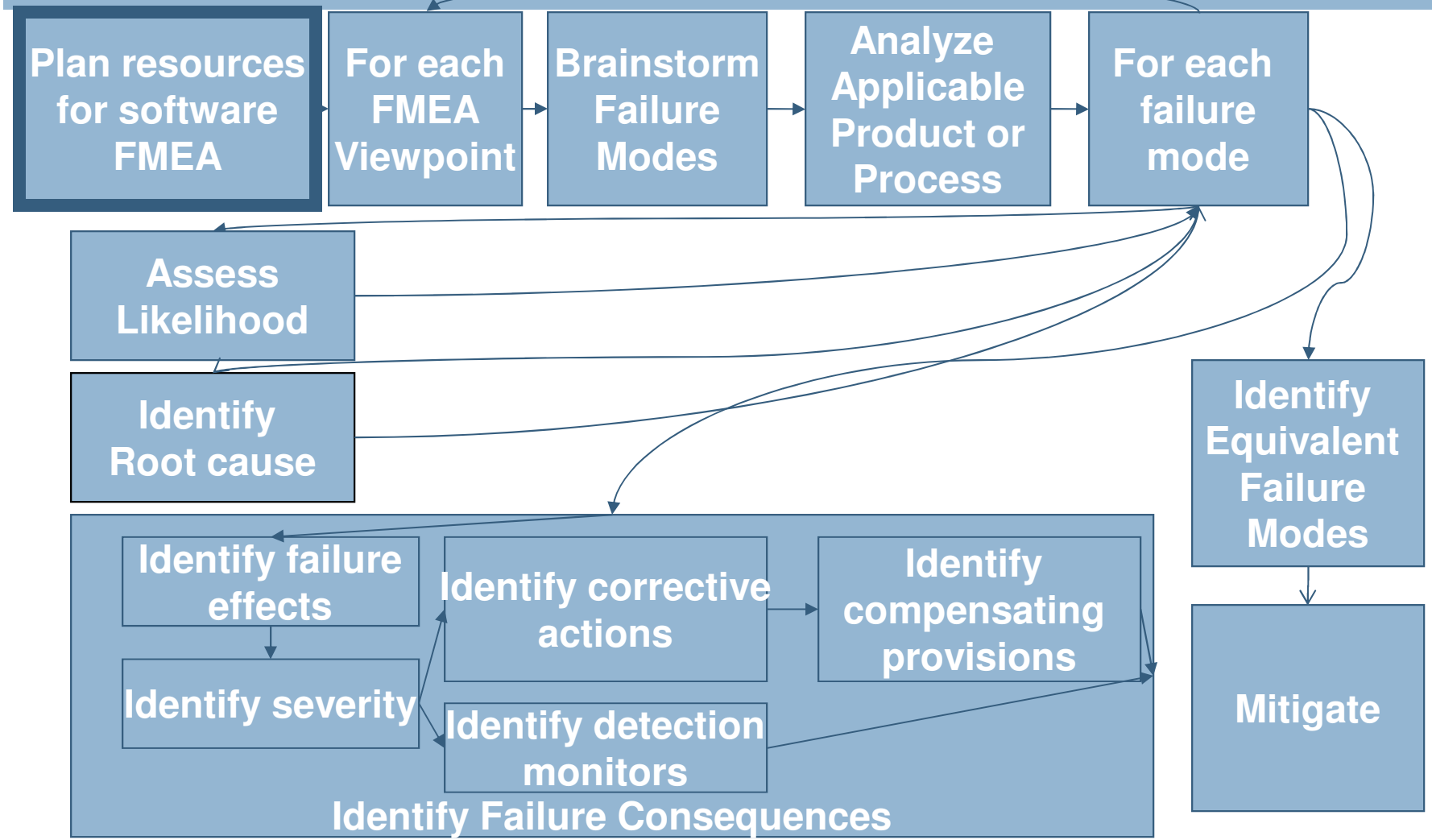
- What does a SFMEA look like?
- What are the steps?
- What are some of the failure modes and root causes?

What does a SFMEA look like?

- Similar to table used for hardware FMEA
- Software engineers have the most trouble getting the left side of the SMFEA started



The process for doing a Software Failure Modes Effects Analyses



The class covers many root cause/failure mode pairs

*Applicable to most if not all application types

		Number of associated root causes		
Failure mode	Description	Functional	Interface	Detailed
*Functionality	Software does not do behave as stated in the requirement	6		3
*Timing	Events happen too late or too early	2	4	
*Sequence	Events happen in the wrong order	5	1	5
*Faulty Data	Data is corrupt, invalid, incomplete or incorrect	5	11	11
Faulty Error Handling *Erroneous or missing error messages *False alarms	•Wrong message, wrong response when an error is detected •Software fails to detect an error when it should •Software detects a error when there is none	5	9	11
Web based	Failure modes specific to HTML, ASP, .Net, etc.	24		

The class covers many root cause/failure mode pairs

*Applicable to most if not all application types

		Number of associated root causes		
Failure mode	Description	Functional	Interface	Detailed
Database related	Storing, retrieving data from a database file		29	
Network communications	Stale data, no communications		6	
Faulty or incompatible I/O	Incomplete or incorrect I/O		15	6
Faulty logic and ranges	Incomplete or overlapping logic			23
*Incorrect algorithms	Formula implemented incorrectly for some or all inputs			8
*Memory management	Out of memory errors			7

The class covers many root cause/failure mode pairs

		Number of associated root causes		
Failure mode	Description	Production	Maintenance	Usage
Execution	Poorly executed project	36		
Tools	Inadequate tools/training/people	15		
Schedule	Inadequate scheduling	23		
Faulty C/A	Change to a correction causes a new defect		See detailed viewpoint	
Unsupportable	Software can't be easily maintained		10	
Unserviceable	Software can't be easily serviced after install		8	
Installation	SW doesn't install/update			23
Human	Human error, misuse or abuse			12
Security	Security violations, overly secure			9
User instructions	Inadequate or conflicting instructions for operating the software			13

General Steps for laying out each SFMEA viewpoint

1. Create one worksheet for each unit that applies for this viewpoint
   - CSCI (functional), module (detailed) or interface pair (interface)
2. Review the product documentation or code associated with the first step
   - SRS (functional), code (detailed), IDS (interface)
3. Create one row for each requirement or data element
4. Review all failure modes related to that view
5. List all of the above failure modes and root causes that are applicable for each row
6. Each row can/will have more than 1 failure mode and/or root cause
7. Once the 4 columns on the left hand side of the table are complete, proceed to the columns on the right side

Example template for the detailed SFMEA

Function	Description	Failure mode	Root causes
Module name	Name of variable, type, size, min, max and default value	Faulty data	List all root causes that apply to this data element
Module name	List each algorithm	Faulty algorithm	List all root causes that apply to this algorithm
Module name	Required logic	Faulty Logic	List all root causes that apply to this logic
Module name	Required ranges	Faulty ranges	List all root causes that apply to the ranges defined by this logic

Example: Some Root Causes of Faulty Range Data Failure Mode for Detailed FMEA viewpoint

1. Module does not work for upper bounds on input variables
2. Module does not work for lower bounds on input variables
3. Module does not work for intersections of input ranges
4. Module defines a > b when there should be a >= b
5. Module defines a < b when there should be a <= b
6. Module defines a >= b when there should be a > b
7. Module defines a <= b when there should be a < b
8. Overflow ignored
9. Improper comparison of variables with 2 different formats
10. Equality Comparison between floating point value and zero

References

- [1] "SAE ARP 5580 Recommended Failure Modes and Effects Analysis (FMEA) Practices for Non-Automobile Applications", July, 2001, Society of Automotive Engineers.
- [2] "Software Systems Testing and Quality Assurance", Boris Beizer, 1984, Van Nostrand Reinhold, New York, NY.
- [3] "A Taxonomy of E-commerce Risk and Failures", Giridharan Vilangadu Vijayaraghaven, A Thesis Submitted to the Department of Computer Science at Florida Institute of Technology, Melbourne, Florida, May 2003.