SOFTWARE FAILURE MODES AND EFFECTS ANALYSIS **(FMEA)**

FMEA for Software? If your system is safety critical, and your hardware is getting the FMEA treatment, you had better not treat your software as any less critical. As in the case of hardware, a software FMEA is an incredibly valuable addition to the organizational knowledge base. Every additional program FMEA will reduce future FMEA efforts and will also provide the basis for safer and more cost effective design and coding in the future.

As in hardware, the software FMEA shows:

- Critical failure effects
- Failure modes leading to these effects
- Where additional protection is required.

Does software fail? We tend to believe that well written, well tested, safety critical software never fails. Experience proves otherwise, with software making headlines when it actually does fail, sometimes critically. Software does not fail the same way hardware does, and the various failure behaviors we are accustomed to from the world of hardware are often not applicable to software. However, software does fail, and when it does, it can be just as catastrophic as hardware failures. The FMEA is not specific to a type of failure behavior or a certain type of failure statistic; it is universal and extremely useful to software as well. When properly done, the FMEA offers an exhaustive and complete review of potential critical failures due to software function.

What are "software failure modes"?

Software, especially in critical systems, tends to fail where least expected. We are usually extremely good at setting up test plans for the main line code of the program, and these sections usually do run flawlessly. Software does not “break” but it must be able to deal with “broken” input and conditions, which are often causes for “software failures”. The task of dealing with abnormal/anomalous conditions and inputs is handled by the exception code dispersed throughout the program. Setting up a test plan and exhaustive test cases for the exception code is by definition difficult and somewhat subjective. The FMEA removes this difficulty and provides a guide to ensure completeness of the testing and certification process.

Anomalous inputs can be due to failed hardware, timing problems, harsh/unexpected environmental conditions and multiple changes in conditions and inputs that are beyond what the hardware is able to deal with. Bad user input may also be a source for such exception conditions. Often the conditions most difficult to predict are multiple, coinciding, irregular inputs and conditions.

How do we protect our critical systems from such software failures?

The FMEA process ensures exhaustive identification of exception condition initiators, and verification that protection against faults in exception handling, are in place and effective!

Although slightly different from a hardware FMEA, when properly executed, the software FMEA is compatible with hardware FMEAs and permits a full system FMEA. Hence it provides the assurance, that other certification processes cannot, that we have identified all possible failure modes and have included provisions to detect and protect against them.

Severity Level:	I	
System Failure	Loss of longitudinal Control	
Item ID	**Item**	**Failure Cause**
1.1.2	Pitch_FB	Stuck
1.1.3.1	<Nz_g>	Stuck
Severity Level:	I	
System Failure	Extreme surface deflection	
Item ID	**Item**	**Failure Cause**
1.1.3.1	<Nz_g>	Absent
Severity Level:	II	
System Failure	Erratic longitudinal control	
Item ID	**Item**	**Failure Cause**
1.1.1	Nz_cmd	> Limit
1.1.2	Pitch_FB	> Limit
1.1.2	Pitch_FB	Absent

A list of system effects according to their severity, and all the failure modes and items that can lead to these effects will provide the backbone to your certification process and will allow complete mitigation of possible safety critical problems.

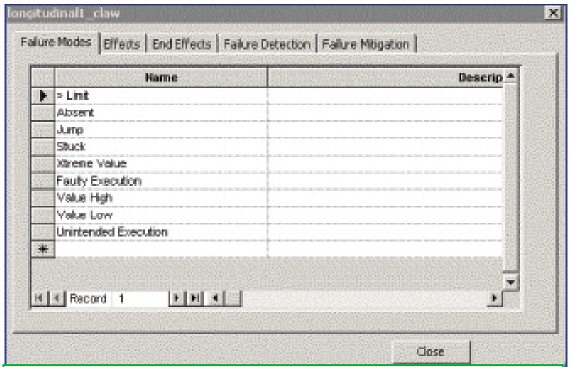
Software FMEA – How?

One of the main reasons the FMEA hasn’t been a consistent part of critical software certification is the difficulty in applying it to a large piece of code. SoHaR has developed a methodology that overcomes this problem by using the object view of the program. Whether developed as a UML or MatLab Simulink model, or coded in an object-oriented language such as C++, .Net or Java, we apply our FMEA methodology at the object level. Along with requirements and design documents we are able to construct a software FMEA

SOFTWARE FAILURE MODES AND EFFECTS ANALYSIS **(FMEA)**

that is surprisingly similar to a hardware FMEA, as software “objects” are equivalent to hardware “parts”. Moreover, when required, we will develop and generate a system FMEA which will include hardware and software and any interface failure modes.

Our method overcomes another inherent software FMEA problem that most professionals cannot escape: the subjectivity of the process. Most software safety professionals will apply the FMEA at a “functional” level. This application is not only problematic in that it can leave entire sections of the exception code unevaluated, but it also introduces a subjectivity into the process that allows more failure modes to be ignored. Our object-centered method removes this subjectivity as it uses the classes defined in the design.



Name	Descrip
> Limit	
Absent	
Jump	
Stuck	
Xtreme Value	
Faulty Execution	
Value High	
Value Low	
Unintended Execution	

SoHaR's automated FMEA tools allow you to build extensive libraries of failure modes, failure effects, system effects, and detection and mitigation provisions. The libraries enrich the organization knowledge base and directly reduce costs in future efforts both by making early designs safer and by reducing future FMEA efforts.

Automated Software FMEA

FMEAs, applied to software or hardware, are a large task. Hardware FMEAs are automated through an exhaustive system breakdown tree, or Bill Of Material. SoHaR has developed automated tools and methods for generating a complete software FMEA based on object-oriented software models. Our tools are currently able to automatically generate the FMEA for models developed in UML (Unified Modeling Language) or within the MatLab Simulink environment. Benefits of using our automated tools include:

- A significant reduction in work load (by several orders of magnitude)
- Assurance of completeness of the task (no failure modes left behind)
- Libraries for future use that reduce work load even more (software and interface components, failure modes, higher order effects, detection methods, compensation provisions)

What Can You Expect From SoHaR’s Software FMEA Services and Tools?

SoHaR provides both consulting services and tools for the Software FMEA. Our services cover the entire spectrum of organizational needs:

- SoHaR can perform the entire task of developing the FMEA for your system and generating the complete FMEA reports.

or

- SoHaR can provide consulting to an in-house effort which may include any combination of: training, system set-up, tools and/or continuous program support.

Either way, SoHaR will walk you through the process so that your organization is able to successfully complete the FMEA and fully trust the results.

What will our FMEA and reports include?

- List of critical failure modes and whether they have been accounted for in the design;
- List of provisions (detection methods & compensation provisions) required to make the current system safe.

At the end of every effort, the reports and electronic libraries developed in the process will lead to an easier task in future FMEA efforts. As in the case of hardware, a software FMEA is an incredibly valuable addition to the organizational knowledge base, allowing for safer and less costly programs in the future.

ALD Headquarters:
Sonol Tower 11th floor 52
Menachem Begin Road,
Tel-Aviv, 67137, Israel
Tel: +972 3 7913200
Fax: +972 3 7913210
E-mail: support@ald.co.il