System Safety Engineering

Dev Raheja
Design for Competitiveness, Inc

System-of-Systems

- This is a central problem in safety management.
- Some pieces are within your organization, most are not!

System-of-Systems

- Is self organizing
  - Cannot understand the system by looking at the pieces only
  - The chain of cause – and – effect is unbounded such as in hospitals
  - Impossible to predict risk with high confidence

Safety specifications imply

- A System-of-Systems
- A system of diverse interactions
- Looking at Big picture to go to the right detail

Therefore…
Tweaking on pieces without the knowledge of whole, is doomed to failure.

Safety Analysis Includes:

- Control of all energies
  - Mechanical in cars
  - Thermal in electronics
  - Pressure in defibrillators
  - Electrical in x-ray
  - Chemical in drugs
  - EMI/RFI in pacemakers

- Analysis Of Hardware, Software, People, Procedures, Tools, Facility, Environment, Communications, And Management!
Managing Complex Risks

Decisions that involve risk avoidance can be made early during the inception of a design.


- FATALITIES: 1.2 million
- COST: 230 Billion
- TRENDS: Overall

Healthcare Safety Mishaps

- As much as 100,000 fatalities each year from medical mistakes
- About 2.1 million patients acquire infections from hospital care
- 40,000 to 80,000 misdiagnosed every year

Cautions

- Software accident trend (↑)
- Active safety is lowering the rate but it is introducing new hazards.
Causes Of Unsafe Systems

- Failure to treat safety as a business case!

Nature of Quantification

- Benchmark against one per million incidences
- Questions:
  - over what time?
  - Is it on incidence frequency?
  - Is it by the number of exposures?

Elements Of A System

- System
  - People
  - Procedures
  - Equipment & Facilities
  - Environment

Boundaries on Safety

- Higher assembly
- OEM system
- User
- Bystander
- Environmental

Hazard Assessment Matrix

<table>
<thead>
<tr>
<th>FREQUENCY OF OCCURRENCE</th>
<th>HAZARD CATEGORIES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 CRITICAL</td>
</tr>
<tr>
<td>A (FREQUENT)</td>
<td>1A</td>
</tr>
<tr>
<td>B (PROBABLE)</td>
<td>2A</td>
</tr>
<tr>
<td>C (OCCASIONAL)</td>
<td>3A</td>
</tr>
<tr>
<td>D (IMPOSSIBLE)</td>
<td>4A</td>
</tr>
</tbody>
</table>

Hazard Risk Index (HRI)
1A, 1B, 1C, 2A, 2B, 3A, 4D, 4E

Management Criteria
1. UNACCEPTABLE
2. UNACCEPTABLE (MANAGEMENT DECISION REQUIRED)
3. ACCEPTABLE WITH REVIEW BY MANAGEMENT
4. ACCEPTABLE WITHOUT REVIEW

When to Perform SHA

<table>
<thead>
<tr>
<th>Hazard Analysis</th>
<th>System Life Cycle Phases</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHA</td>
<td>Planning</td>
</tr>
<tr>
<td>SSA</td>
<td>Planning</td>
</tr>
<tr>
<td>O&amp;S/HSA</td>
<td>Planning</td>
</tr>
<tr>
<td>SA</td>
<td>Planning</td>
</tr>
<tr>
<td>FTA</td>
<td>Planning</td>
</tr>
</tbody>
</table>

PHA = Preliminary Hazard Analysis
SSHA = Subsystem Hazard Analysis
SHA = System Hazard Analysis
O&S/HSA = Operating & Support Hazard Analysis
SSA = Software Safety Analysis
SA = Security Analysis
FTA = Fault Tree Analysis
Trade-off of Hazard Management

- Probability of occurrence
- Severity
- Cost
- Design to eliminate the hazard
- Add Safety/Security devices
- Add warning devices
- Institute special procedures or training

Safety in Design

- Concept validation for safety before approving specifications
- Safety specifications usually miss about 50% requirements.
  - *Start with the assumption that the specs are wrong!*

Compartmentalizing the system for risk reduction

- Separate safety related software from the main software
- Isolate safety related hardware from multiple uses
- Separate safety related maintenance tasks from other tasks
- Allocate responsibility these to those who have through training in safety

Safety in Design

- Precedence For Safety
  - Avoid accident
  - Tolerate the fault
  - Design for limp home safely
  - Design early warnings
  - Develop robust training

Hierarchy Of A System
**Theory of accidents**

- Old Theory
  - Domino effect
  - Single point
  - Human error
  - OR relation
  - Energy

- Current Theory
  - Multiple causes
  - AND relation

**Redefining the System**

- Holistic Considerations
- Life cycle considerations
- Abuse/misuse considerations
- Robustness considerations

**Redefining the System**

- Holistic Considerations
  - Identify “What” the system must always do
  - Identify what the system should “Never” do
  - Safe life = 3 x Normal Life
    (cheaper to do it this way!)

**Redefining the System**

- Life Cycle Considerations
  - Inherent design risks
  - Latent manufacturing defects
  - Supplier risks
  - Testability
  - Repair and maintenance risks

**Redefining the System**

- Robustness Considerations
  - Emergency response
  - Response to user error
  - Response to interface failures
  - Response to unexpected failures

**Fine-tuning Specifications**

- Preliminary Hazard Analysis (PHA) on Preliminary Design
- Identify critical hazards
- Assess risk
- Mitigate risk with safety requirements
- Lower costs
**Fine-tuning Specifications**

- Hazard Analysis
  - Identify possible unsafe conditions
  - Assign severity (I,II,III,IV) and frequency(A,B,C,D,E)
  - Use robust safety criteria

- Cost Effective Controls
  - Set high ROI target, even on safety
  - Be creative. *You can do it if you believe in yourself*

**Subsystem Safety Analysis**

- Hazards
  - Blown tire
  - Missing screw
  - Changing battery

- Trigger
  - SUV unstable
  - Aircraft fails
  - User makes mistake

**Subsystem Hazard Analysis (SSHA)**

- To prevent accident
  - Prevent hazard, or
  - Prevent trigger event
  - Update specification

**Minimizing Accidents in Early Design**

- Conceptual completeness
- Logical solutions
- Intuitive solutions
- Innovation with high ROI

**Methods of Assessing Risk**

- Probabilistic risk analysis
  - least credible
- Quantitative dollar risk analysis
  - More credible
- Intuitive risk analysis
  - Less credible but more real
Service Safety Analysis

- Write Maintenance Procedure
- Identify hazards
- Identify trigger events
- Prevent either one
- Update specification

Minimizing Accidents in Detailed Design

- The more tools used, the safer the design
- Use Fault Tree Analysis (FTA)
- Use Failure Mode & Effects Analysis (FMEA)
- Use Operating & Support Hazard Analysis (O&SHA)

Using Fault Trees to Mitigate Risks

- No single point failures leading to top events allowed
- No common hardware, software or procedures for redundant paths
- Choose top down structure for the system similar to fault trees
- Analyze the control loop to make sure the feedback is never lost

Mitigating Latent Hazards and Faults In Production

- Perform process hazard analysis
- Qualify production for absence of hazards
- Use feedback from production and field
- Design to avoid latent defects

FMEA At Subsystem Level

Risk Verification and Validation

- Documents Required
  - Preliminary Hazard Analysis (PHA)
  - Subsystem Hazard Analysis (SSHA)
  - System Hazard Analysis (SHA)
  - Service Safety Analysis
  - Operating & Support Hazard Analysis (O&SHA)
Verification and Validation

- Test For
  - Response to all failures
  - Response to rare event
  - Response to wrong inputs
  - Response to wrong outputs
  - Response to Alerts/Alarms

Embedded Software Safety

- Examples of software hazards
  - Audi 5000
  - Sudden acceleration on highway
  - Pacemakers

Workshop: Testing for Safety

- Identify a failure
- Identify a rare event
- Identify a wrong input
- Identify a wrong output
- Develop response for each condition

Increase Value in Testing

- Develop test to prove how good the safety is—not just meeting the specifications
- Test to discover mistakes in design and in production

Perform Software Hazard Analysis

- At Functional Level
- At Logic Level
- At Code Level

Writing Software Requirements

- Software requirements must be derived from the system specification
- Identify what software functions support the system functions
- Make sure the response to each failure is failsafe
Requirements Analysis

- Make sure each requirement is measurable
  - Ask: What can go wrong?
- Identify rare events and assure fail safe response
  - Ask: What are the stupid things the software should never do?

Software Safety Design Control

- Select right architecture: Top down structure recommended
- Select right architecture: Fault tolerant architecture recommended

Software Safety Design Control

- Other techniques
  - Use checklists
  - Defensive programming
  - Independence of modules
  - Input/output boundary validation

Conclusions…

- The whole is more than the sum of its parts.
- If You Look Beyond The Box, You Will See Dramatic Savings From Safety.
- We can make safety a business case instead of cost!