Evaluating hazards in critical software-dependent systems

Sushil Birla
U.S. Nuclear Regulatory Commission
ASQ Section 509, December 15, 2015
Outline

• The problem in focus
• Software quality – knowledge evolution
• Quality models – framework
• Streamlined quality model to evaluate safety
• Derivation of constraints
• Related R&D activities
• Reasoning structure
• Future directions
Focus

• Hazards in a critical cyber physical system
  – contributed through systemic causes, e.g., engineering deficiencies.

• Integrated approach to:
  – systems engineering
  – software engineering
  – safety engineering
Hazard analysis in relation to Safety Analysis (SA)

Hazard Analysis (HA)
Result: Safety Requirements

Verification (V)
Result: Evidence of Requirements met

Safety goals

Integration of evidence (V) to show that Safety goals are met.
Identify safety goals, requirements & constraints through Hazard Analysis

Losses to be prevented by this system are identified correctly

All (direct) hazards are identified

Required control actions are formulated

All contributory hazards are identified

Verifiable requirements and constraints are formulated
Mishaps in software-dependent critical systems even after

- The system was verified to provide all the needed functions
- All constituents of the system met their respective requirement specifications
- None, by itself, “failed.”
A weak link in requirements engineering

V&V

Verification
“Have I met the requirements?”

Validation
“Do I have the right requirements?”

100% Verification will never identify inadequate requirements

Weakest Link is establishing valid requirements to drive system design
Aspect of the gap in focus

- Requirements
  - Functional Requirements
  - Quality Requirements
    - Safety
    - Security
    - Other
Quality models: framework

Quality goals

define stakeholders’ min. acceptable levels of quality for a System

specify achievement of Quality requirements

shall exceed Quality thresholds

are measured along using

are stated in terms of Quality characteristics

are stated at Quality attributes

apply during Conditions

shall exceed System-specific Quality criteria

are measured using Measurement scales

define the meaning of quality of a

define the meaning of quality of a

Quality requirements

Measurement methods

Measurement scales

Quality attributes

Quality characteristics

Events

Conditions
Streamlined Quality model to evaluate safety

Comprehensibility

Complexity

Simplicity

Assurability

Verifiability

Analyzability

Predictability

Safety

Freedom from interference

Deterministic behavior
HA: Source of validated safety requirements

Information from next higher integration level

HA\textsubscript{T1-T4} → Plans. Concept → Derived requirements → Architecture → Detailed design → Implementation →...

Safety engineering

Development engineering

Verification

V\textsubscript{requirements} → V\textsubscript{architecture} → V\textsubscript{DetailedDesign} → V\textsubscript{implementation} → V...
Current State & Trends

Trends
- Interconnections ↑
- Feedback paths ↑
- Complexity ↑
- Comprehensibility ↓
- Verifiability ↓
- Analyzability ↓
- Deterministic behavior ↓

Side effects
- Unwanted interactions ↑
- Hidden dependencies ↑
- Independence ↓
- Redundancy ↓
- Defense in depth ↓
- Diversity ↓
- Safety margins ↓
- Common cause ↑

Consequence
- Traditional HA techniques (FTA; DFMEA) ineffective
  [RIL-1001; RIL-1002; NUREG/IA-0254; RIL-1101]
• Since 1976: R&D on quality frameworks, models:
  – McCall; Boehm; U.S. Air Force
• 1979: F. Roberts – Measurement theory ...
• 1978-1985: ISO/IEC JTC1 efforts
  – Developed consensus for common standard
• 1994: QM-QA Vocab superseded by ISO 9000
• 2001: ISO/IEC 9126
  – State of the art inadequate.
  – Quality characteristics from ISO 84021.
    • Used broadly; Starts with the user's needs.
• 2005-2011: ISO/IEC 25000 series SQuaRE
• 2011: ISO/IEC/IEEE 29148 - RE
Some related R&D activities

- **Boehm, USC:**
  - Tradeoffs across competing quality attributes
- **Software Engineering Institute**
  - Quality attributes → Architecture
  - Integrated assurance & development environment
    - DARPA HACMS Rockwell Collins project
- **NRC RIL-1101** [http://cps-vo.org/node/8758](http://cps-vo.org/node/8758).
Each anomaly or uncertainty by itself seems to be small.
Structure or structures of the system, which comprise elements, the externally visible properties of those elements, and the relationships among them and with the environment.

WHERE:

**System:** combination of interacting elements organized to achieve one or more stated purposes. Systems can comprise of systems. A system with only software elements is also a system.

**Environment:** includes the combination of systems and elements external to this system, human elements interacting directly with the system and the commensurate manual procedures.

**Element:** a discrete part of a system that can be implemented to fulfill specified requirements. Examples: hardware, software, data (structure), human, process (e.g., process for providing service to users), procedure (e.g., operator instructions), facility, materials, or any combination.

**Externally visible properties:** include behavior – normal, as well as abnormal.

**Relationships:** include interactions and interconnections (communication paths).
Framework to integrate evidence

Some major sources of uncertainties

- Environment
  - Assumptions
  - Input validity
- Requirements
  - Correct?
  - Complete?
  - Consistent?
- Incomplete coverage
- Interference

Coverage evidence (Diverse complementary):

- Analysis
- Model checking
- Testing
  - Coverage based
- ...
Loss of concern due to:

- Unrecognized inter-dependencies in the system
- Unrecognized inter-dependencies in processes:
  - Technical
  - Supporting
  - Management
- Disruption in or emissions from the environment
- Emissions/outputs from or behavior of the system
- Unrecognized anomaly in the state of the process (reality ≠ perception)
- Ill-understood nature of change in monitored phenomenon
- Unrecognized anomaly in the state of the instrumentation
- Unrecognized anomaly in state of some element in environment
- Ill-defined boundary of the system
- Unrecognized hazards from interactions of the system with its environment
  - e.g., effects of invalid assumptions,
Hazard scenario contributed through non-verifiability:
Appropriate criteria are not formulated at beginning; therefore, corresponding architectural constraints are not formalized and checked. By the time work products are available for testing, it is discovered that adequate testing is not feasible.

Examples of derived constraints:
• Verifiability is required at all levels in the system integration
  – down to smallest element
• Unnecessary complexity is avoided
• Behavior is unambiguously specified
• System behavior is composition of behaviors of its elements.
  – No unpredictable behavior emerges.
  – Then, if all elements are verified individually, their composition is also considered verified.
• Interactions precluding complete verification are not
Examples of hazards contributed through interference:
• Connection across lines of defense blurs the lines
• Connection across redundant elements compromises redundancy
• Summing of signals at voting logic compromises their sources
• A monitor is compromised by the monitored element
• A safety function is compromised by a non-safety element

Examples of derived constraints:
• Freedom from interference is assured provably across:
  – Lines of defense
  – Redundant divisions of system
  – Degrees of safety qualification
  – Monitoring & monitored elements of system
• Interactions limited provably to those required for safety functions
• Interactions and interconnections that preclude complete verification are avoided, eliminated, or prevented
Hazard scenario contributed through non-comprehension:
System behavior is not understood in the same way by its community of users...

Examples of derived constraints:
- Behavior is understood completely, correctly, consistently, and unambiguously by different users interacting with the system
- The allocation of requirements to some function and that function to some element of the system is bi-directionally traceable.
- The behavior specification avoids mode confusion, esp. when functionality is nested
- The architecture is specified in a manner (e.g., language; structure) that is unambiguously comprehensible to the community of its users, e.g. reviewers, architects, designers, implementers, etc., i.e. the people and the tools they use.
From characteristics to constraints
Flow-down - 1/5

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Derived constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verifiability</td>
<td>Required property, flowing down to the finest-grained element. Verifiability checked at every phase of the development lifecycle, at every level of integration, before proceeding further. Examples of conditions for verifiability:</td>
</tr>
<tr>
<td></td>
<td>• Ability to create a test (or verification) case;</td>
</tr>
<tr>
<td></td>
<td>• Observability;</td>
</tr>
<tr>
<td></td>
<td>• Ability to constrain the environment.</td>
</tr>
</tbody>
</table>

Evidence of verifiability in verification plan.
<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Derived constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyzability</td>
<td>Avoidance of unnecessary complexity.</td>
</tr>
</tbody>
</table>

Behaviour unambiguously specified

Flow-down is correct;
1. Allocated behaviors satisfy behavior specified at next higher level of integration;
2. Unspecified behavior does not occur.

System behaviour = ©{behaviours of its elements}

Development follows a refinement process.
## From characteristics to constraints 3/5

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Derived constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unanalyzed or un-analyzable conditions</td>
<td>System is statically analyzable.</td>
</tr>
<tr>
<td></td>
<td>1. All states, including fault conditions, are known.</td>
</tr>
<tr>
<td></td>
<td>2. All fault states, leading to failure modes, are known.</td>
</tr>
<tr>
<td></td>
<td>3. Safe state space of the system is known.</td>
</tr>
<tr>
<td>Characteristic</td>
<td>Derived constraint</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>System behaviour</td>
<td></td>
</tr>
<tr>
<td>deterministic</td>
<td>System has a defined initial state.</td>
</tr>
<tr>
<td></td>
<td>System is always in a known configuration.</td>
</tr>
<tr>
<td></td>
<td>System is in a known state at all times.</td>
</tr>
<tr>
<td>System behaviour</td>
<td>Each state transition is specified and known, including transitions corresponding to unexpected conditions.</td>
</tr>
<tr>
<td>predictable</td>
<td>A hazardous condition can be detected in time to maintain the system in a safe state.</td>
</tr>
<tr>
<td>Characteristic</td>
<td>Derived constraint</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Comprehensibility</td>
<td>Behaviour is completely and explicitly specified.</td>
</tr>
<tr>
<td></td>
<td>Behaviour is completely understandable.</td>
</tr>
<tr>
<td></td>
<td>Behaviour is understood completely, correctly, consistently, and unambiguously by different users interacting with the system</td>
</tr>
<tr>
<td></td>
<td>The allocation of requirements to some function and that function to some element of the system is bi-directionally traceable.</td>
</tr>
<tr>
<td></td>
<td>No mode confusion, esp. when functionality is nested.</td>
</tr>
<tr>
<td></td>
<td>Architecture spec. is unambiguously comprehensible to the community of its users - people and the tools they use.</td>
</tr>
</tbody>
</table>
Reasoning structure

Causal model

Inference rule

Evidence

Reasoning

Factors influencing validity

Assertion/Claim

Qualifiers:
- Degree of validity
- Conditions/Deficits

Challenges; rebuttals; inconsistencies

basis for

used in

affects
Questions/Discussion
Other Information for Reference
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DARPA</td>
<td>Defense Advanced Research Projects Agency</td>
</tr>
<tr>
<td>DI&amp;C</td>
<td>Digital Instrumentation and Control</td>
</tr>
<tr>
<td>EPRI</td>
<td>Electrical Power Research Institute</td>
</tr>
<tr>
<td>HACMS</td>
<td>High Assurance Cyber Military Systems</td>
</tr>
<tr>
<td>I&amp;C</td>
<td>Instrumentation and Control</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electrotechnical Commission</td>
</tr>
<tr>
<td>IEEE</td>
<td>International Electrical &amp; Electronics Engineering Society</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standards Organization</td>
</tr>
<tr>
<td>JTC1</td>
<td>Joint Technical Committee 1</td>
</tr>
<tr>
<td>min.</td>
<td>minimum</td>
</tr>
<tr>
<td>NPP</td>
<td>Nuclear Power Plant</td>
</tr>
<tr>
<td>NRC</td>
<td>Nuclear Regulatory Commission</td>
</tr>
<tr>
<td>Acronym</td>
<td>Definition</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>QA</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>QM</td>
<td>Quality Management</td>
</tr>
<tr>
<td>R&amp;D</td>
<td>Research and Development</td>
</tr>
<tr>
<td>RE</td>
<td>Requirements Engineering</td>
</tr>
<tr>
<td>RES</td>
<td>NRC Office of Nuclear Regulatory Research</td>
</tr>
<tr>
<td>RIL</td>
<td>Research Information Letter</td>
</tr>
<tr>
<td>spec.</td>
<td>specification</td>
</tr>
<tr>
<td>SQuaRE</td>
<td>Software Quality and Requirements Engineering</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>USC</td>
<td>University of Southern California</td>
</tr>
<tr>
<td>V&amp;V</td>
<td>Verification and Validation</td>
</tr>
<tr>
<td>Vocab</td>
<td>Vocabulary</td>
</tr>
</tbody>
</table>
Intent, needs, requirements, specifications, procedures, constraints

Incoming item, e.g. work product of preceding phase

Process activity

Work Product

applied to

Resources

Human

Tools

Aids

Information

Others
Key factors affecting HA Quality

Organizational culture, context, processes, constraints, etc.

Relevant results from next higher level of integration

Result of HA, e.g., constraints on item, plans, processes, etc.

Resources

Human

Tools

Aids

Information

Others

\( HA_{T1-T4} \)