Measuring Complexity for System Safety Assurance

Sarah Sheard, Mike Konrad, Bill Nichols, Charles B. Weinstock

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213
Purpose

Question: When is a system is too complex to certify as safe?
Possible Solution: Error propagation complexity algorithm
Background

2014: FAA requested research on system complexity and safety, including definition and measurement

Requested avionics-specific definitions of complexity and complexity measure(s)

What threshold of that measure might make a system too complex to be able to assure safety?

Funded SEI research project

Output is Final Report and 5 white papers (Complexity overview, Candidate Measures, Safety Cases, Complexity Calculation Algorithm, Algorithm Test)
Complexity is complex

What is “Complexity

What is complex?

How complex is it?

Cyclomatic Complexity
Fan-out and Fan-in
Requirements Churn

What about Complexity matters to Safety?

Software
Hardware
Avionics
Plane?
Requirements
Designs
Models
Tests
...?

Size (number)
Diversity or Variety
Diversity of Relationships / Interconnections
Relationships / Interconnections

Software Solutions Symposium 2017

Measuring Complexity for System Safety Assurance
March 20–23, 2017
© 2017 Carnegie Mellon University

[Distribution Statement A] This material has been approved for public release and unlimited distribution.
Safety Case (type of Assurance Case)

For “The System Is Safe” to be true
- Subclaim 1 and 2 must be true
- Argument must be sound

For Subclaim 1 to be true
- There must be X evidence

For Subclaim 2 to be true
- Subclaim 3 and 4 must be true
  - For Subclaim 3 and 4 to be true
    - There must be Y evidence
    - There must be Z evidence

Multiple technical exchange meetings
2 Breakthroughs

1. Evaluate the complexity *of the safety case*

But: the safety case isn’t “complete” until the aircraft is designed, built, tested, with all software on board…

2. **Estimate** the size of the safety case **early**

How much work (analysis, documentation, meetings etc.) will it take to prove the system is safe?

(# potentially cascading error conditions)

- Assume component assurance process will remain as is
- Big open question is errors cascading from one component to another
- Order of magnitude probably ok
Our Method

Primary Assumption:

Early design work on new system* has resulted in a model of the system architecture at a high level including

• system modes
• active components and their interconnections in each mode
• possible failure conditions that could propagate outward

Many additional assumptions made to arrive at notional thresholds for between systems that are assurable as safe and systems that are too complex to assure as safe

*For future research: precedented systems
Assume

Multiple modes; errors can propagate in each
  ► Sum over all modes
Multiple components; errors can propagate from each one
  ► Sum over all components active in that mode
Multiple propagation points on components
  ► Sum over all (outward-) propagation points

Then,

For each propagation point, each component, each mode:
  ► Multiply number of failures that could propagate out by number of places the failures could reach (Fanout)
Algorithm

Sum over all system modes:

Sum over all components active in a given mode:

Sum over all propagation points (p-points) for this component:

\[
\begin{align*}
\text{Number of failures that could propagate} & \quad \text{times} \quad \text{Fanout from this p-point} \\
\text{out from this p-point} & 
\end{align*}
\]
Example 1: Stepper Motor System

1. From High Level design:
   - 1 mode
   - Interfaces shown
   - Treat Bus 2 as a component*
   - 4 components plus Environment
   - #P-points = 1 for all components
   - Fanout always = 1

2. From Error Model:
   - Errors from Environment to SMS: 3
   - Errors from PCS to Bus 2: 4
   - Errors from Bus 2 to ACT: 3
   - Errors from ACT to motor: 3
   - Errors from Motor to Envt.: 3

*Since it can be a source of a failure condition

Ref: Konrad 2015b of Final Report
Calculating EPC (for one mode)

First step

- Env 1
  - P(1,1)*

- PCS 2
  - P(2,1)

- Bus2 3
  - P(3,1)

- ACT 4
  - P(4,1)

- Motor 5
  - P(5,1)

*Notation P(component#, p-point#)

Second step

- Env 1
  - 3

- PCS 2
  - 4

- Bus2 3
  - 3

- ACT 4
  - 3

- Motor 5
  - 3

Third step

Sum of (#failures*Fanout for all P-points of Component x)

<table>
<thead>
<tr>
<th>x</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3*1 = 3</td>
</tr>
<tr>
<td>2</td>
<td>4*1 = 4</td>
</tr>
<tr>
<td>3</td>
<td>3*1 = 3</td>
</tr>
<tr>
<td>4</td>
<td>3*1 = 3</td>
</tr>
<tr>
<td>5</td>
<td>3*1 = 3</td>
</tr>
</tbody>
</table>

Total all components

$Error\ Propagation\ Complexity = 16$
Potential Applications of This Research

• FAA uses as evidence that they need to ask manufacturers to provide documented safety cases rather than just standards compliance

• Manufacturers (1st and lower tiers) use estimate of design complexity to estimate their own QA effort

• Comparison of designs by how complex are their error propagation potentials

• Complexity as an indicator of risk, to be tracked using standard techniques

• Future research into “how much can we discount the complexity of a system given that X% has been used before?” can be framed as “Credit for Precedence” and ties to “Recertification” questions. Much interest across SEI and at CMU for this topic
Contributions

First tie of system complexity to safety that we know of

Use Safety Case review time estimate as a proxy for complexity

With architecture model, program, can estimate complexity of
different alternatives as they will relate to safety, and can compare them
Recommended Future Research

1) Apply and validate to larger system at real-life scale.

2) Study special cases, assumptions, and limitations more specifically
   a) Including what about preceded system components: should these count as less complex because we are familiar with them? How?
   b) Including tweak numbers for whether the Applicant has provided an organized assurance case or not. How does this affect FAA effort?
   c) Determine effect of having models to different levels of detail. Is there a notional “complexity reduction” curve?

3) Expand fault model to include more than error propagation: emergent behavior, concurrency, and cybersecurity

4) Develop guidelines for safe assurance practices and design guidelines to reduce software complexity
For More Information: Report and White Papers

http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=483758

Report:


White Papers


## Contact Information

<table>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>Office Contact</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sarah A. Sheard, Ph.D.</td>
<td>Principal Engineer</td>
<td>(412) 268-7612</td>
<td><a href="mailto:sheard@sei.cmu.edu">sheard@sei.cmu.edu</a></td>
</tr>
<tr>
<td>Michael D. Konrad, Ph.D.</td>
<td>Principal Researcher</td>
<td>(412) 268-5813</td>
<td><a href="mailto:mdk@sei.cmu.edu">mdk@sei.cmu.edu</a></td>
</tr>
<tr>
<td>Charles B. Weinstock, Ph.D.</td>
<td>Principal Researcher</td>
<td>(412) 268-7612</td>
<td><a href="mailto:weinstock@sei.cmu.edu">weinstock@sei.cmu.edu</a></td>
</tr>
<tr>
<td>William R. Nichols, Ph.D.</td>
<td>Senior Member, Technical Staff</td>
<td>(412) 268-1727</td>
<td><a href="mailto:mdk@sei.cmu.edu">mdk@sei.cmu.edu</a></td>
</tr>
</tbody>
</table>

Software Engineering Institute  
Carnegie Mellon University
#1 Recommended Future Research: Precedence

- Study complexity “discounts” that we should give to known or precededent system components because they are familiar
  - How many error propagations (from model) have already been proven not to be unsafe and thus need less review?
  - How can this be applied to, say, *slightly* different configurations? How do you measure “slightly”?
  - How can this be applied to slightly different hazards?
  - What is safety effect of higher-capability component compared to existing?
- Other areas can contribute:
  - How organizations today currently allow credit for testing already done
    - FAA and aircraft re-certification (e.g. longer fuselage)
    - FDA and medical devices
    - Regression testing
  - Estimate of the amount of impact caused by a change (hardware, then software)
  - Understanding how much of the problem could be solved by nearly-independent, modularized, proven-correct components