

Safety Critical Open Systems?

D. Emery

emery@mitre.org

What is Safety-Critical Software?

- Software that contributes to the function of a system where a failure of the system can cause a risk to human life
- Software developed under the provisions of a systems safety program, such as
 - DO-178b (avionics)
 - FDA 247 (medical devices)
 - ANS 7.4.3.2 (nuclear powerplants)
 - Mil-Std 882d (weapon systems)

Characteristics of a Safety Program

- The System has hazards (e.g. "airplane engines stop in mid-flight")
- Hazards have consequences ("airplane proceeds to fall like a rock, killing all aboard")
 - Most safety standards identify severity levels of hazards
- Hazards require mitigations ("prevent engines from stopping")
 - Mitigation strategy based on severity level of the hazard
- System is certified that hazards have been mitigated

DO-178b as an example

- 5 Levels of Failure Condition
 - Ranging from A - Catastrophic, to E - No Effect
- 5 Levels of software effects
 - Ranging from 1 - causes Level A failure, to 5 - No Effect on operational capability or pilot workload

DO-178b defines processes

- Software Planning
- Software Development
 - Requirements, Design, Coding, Integration
- Software Verification
- Software Configuration Management
- Software Quality Assurance
- Certification Considerations

Software Verification

- "Detect and Report errors introduced during the development process".
 - "System requirements properly allocated to software requirements"
 - "High-level requirements decomposed into lower level requirements [including derived requirements]"
 - "Requirements developed into source code"
 - "Executable object code implements requirements"
 - "Means used to satisfy these objectives are technically correct and complete for the software level"
- Software Testing
 - "Software satisfies requirements"
 - "Errors leading to unacceptable [hazards] have been removed"

"Traditional" Development

- Performed by prime and subs according to rigorous, predefined process and procedures
 - All code developed from scratch
 - Processes pre-defined to facilitate safety verification approach
- Reuse limited to code developed by the team under the same/similar procedures
- Higher levels of software more restrictive
 - E.g. no allocated storage allowed at level 1

Certification

- Inspection on the full set of processes
 - Developer proposes a "means of compliance"
 - Certifier approves "means"
 - Developer provides evidence of conformance to "approved means"
 - Certifier reviews evidence

Two key cost drivers for safety-critical software

- Cost to develop
 - Rigor in design process, traceability
 - Impact of restrictions on language features
- Cost to verify/certify
 - Testing, other verification measures
 - Certification documentation

Impact of COTS

- Common avionics developer term for COTS: SOAP
 - Software of Unknown Pedigree
- "Pedigree" includes development process, test process, verification process, CM process
 - Bottom line to prime contractor is ability to certify the system (as part of delivery)

The Challenge

- How to achieve cost savings via reuse of COTS, without compromising safety
 - Reduce development costs
 - Reduce test and certification costs

Reminder

- Safety is a property of Systems, **not** Software
 - Software can perform correctly and system is unsafe
 - Software can perform incorrectly and system is safe

"Safety-critical software" is software potentially suitable for use in a safety-critical system. Without system hazards, no proof that the software itself is "Safety-critical"...

What can COTS developers do?

- Design/develop applications that meet development process requirements
 - Test applications that meet test process requirements
- Provide documentation and other assistance to verification process

"Openness" for Safety

- "Open" means that a piece of software is potentially suitable for use in a variety of safety-critical systems
 - In particular, across system safety standards
- "Open" also means that a piece of software can potentially be replaced by a functionally equivalent piece of software
- Finally, "Open" means that appropriate access is provided to software and documentation

A Potential OpenGroup Agenda

- A common model of development practices
 - E.g. what is required at each level of DO-178b, FDA 247, etc
 - No consensus within the system safety standards community to date...
- Common model of artifacts for verification
 - E.g. Documentation, test reporting, source code access
- Common model of COTS vendor support for certification activities

Advantages & Disadvantages

- + Costs of development & certification spread across all users of COTS products
- + Increased competition, larger market
 - Number of software-intensive safety-critical systems growing substantially...
 - Applicability of safety techniques to other 'mission-critical' aspects of systems
- Increased risks for primes due to less control over COTS vendors
- Less understanding of behavior of software in a specific system

Should the Forum accept
this challenge?