Safety Critical <u>Open</u> Systems

David Emery
emery@davebert.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, the higher the hazard, the more rigorous the mitigation requirement
- 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"

DO-178b section 6

"Traditional" Safety Critical Software Development

- Performed by prime and subs according to rigorous, predefined process and procedures
  - All code developed from scratch within existing vendor team arrangements
  - 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 (DO-178b approach)

- Inspection on the full set of processes
  - 1. Developer proposes a "means of compliance"
  - 2. Certifier approves "means"
  - 3. Developer provides evidence of conformance to "approved means"
  - 4. 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
    - ~$3k to execute each test
  - 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/system integrator is ability to certify the system (as part of delivery)
  - Risk/uncertainty is A Bad Thing

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

Open Group RTE Forum Product: Safety Critical COTS Recommended Practice

- “Reference model” for COTS integration into safety critical systems
- Common model of artifacts for verification
  - E.g. Documentation, test reporting, source code access
  - Based on XML capturing content of data (but not format of data)
- Common model of COTS vendor support for certification activities

Advantages & Disadvantages (for a R.P. for safety critical COTS)

+ 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

Status of the R.P.

- Acceptance within Open Group RTE Forum
  - Briefed at both US and European meetings
- Acceptance and interest in larger safety critical community
  - Briefed at US "Software Safety Working Group"
- Draft White Paper on COTS in Safety Critical Systems
- Informal agreement to reuse existing XML for safety critical documentation

Now we need to start writing the document!

References: Example Safety Standards

- (Commercial) Avionics: DO-178b, DO-248a
- Medical Devices: FDA 247, FDA 351 draft
- Nuclear Power: NRC Guide 1.173 , ANS 7-4.3.2
- Weapon Systems: Mil-Std 882d, DefStan 00-55, 00-56, 00-58
- Electric/Electronic/Programmable Systems: IEC 61508 (series)