Architecture, Arguments, and Confidence

(Joint work with Bev Littlewood, City University, London UK)

John Rushby

Computer Science Laboratory
SRI International
Menlo Park CA USA

Overview

- Many assurance cases involve quantification of risk
- Which in turn requires quantifying failure rates of software
- Notoriously hard to do, beyond about 10^{-3}
    - Which you can test for
- So to provide assessments for higher reliabilities, either need very strong analysis
    - Viewed skeptically by some: e.g., CAST 24
- Or software redundancy
- And that requires choices about the software architecture, the kinds of claims, and the types of argument that can support an assurance case that involves software redundancy

Overview (ctd.)

- I'll outline an approach that combines consideration of architecture, claims about formal verification, and novel probabilistic reasoning
- Will apply it first to one-out-of-two architectures of the kind used for nuclear shutdown
- Then to monitored architectures of a kind proposed for aircraft (software IVHM)

Reliability of Redundant and Monitored Systems

- It is well-known that the reliability of systems with redundant software channels cannot be estimated simply by multiplying the reliabilities of their constituent channels
- Empirical and theoretical studies confirm that failures may not be independent
  - Even when channels are deliberately diverse
  - Some situations are intrinsically more difficult
- Littlewood and Miller model gives probability of system failure as $pfd_A \times pfd_B + Cov(\theta_A, \theta_B)$ where θ_A, θ_B are the difficulty function random variables for the two channels
- Hard to estimate these, and their covariance
- Same considerations apply when we have an operational (sub)system and a monitor

Reliability of Systems With a Possibly-Perfect Monitor

- But suppose the claim we make for the monitor is not that it achieves some particular reliability
    - i.e., has some probability of failure on demand
- But that it is possibly perfect
    - Will need to be simple, and have very strong assurance
- Perfect means that it will never experience a failure
- Possibly perfect means there is some uncertainty about its perfection
    - In particular, it has a probability of imperfection
- We need to be careful about the uncertainties and probabilities here

Aleatory and Epistemic Uncertainty

- Aleatory or irreducible uncertainty
  - is “uncertainty in the world”
  - e.g., if I have a biased coin with $P(heads) = p_h$, I cannot predict exactly how many heads will occur in 100 trials because of randomness in the world

  Frequentist interpretation of probability needed here

- Epistemic or reducible uncertainty
  - is “uncertainty about the world”
  - e.g., if I give you the biased coin, you will not know p_h; you can estimate it, and can try to improve your estimate by doing experiments, learning something about its manufacture, the historical record of similar coins etc.

  Frequentist and subjective interpretations OK here

Aleatory and Epistemic Uncertainty in Models

- In much scientific modeling, the aleatory uncertainty is captured conditionally in a model with parameters
- And the epistemic uncertainty centers upon the values of these parameters
- As in the coin tossing example

One Out Of Two (1oo2) Architectures

- These are systems, like those used for nuclear shutdown, that have two dissimilar channels in parallel
- Either can shut the system down (no voting)
- So system failure requires both channels to fail
- Suppose one is a complex, but highly reliable system A, with aleatory probability of failure on demand (pfd) p_A
- And suppose the other is a simple system B that is possibly perfect with aleatory probability of imperfection (pnp) p_B
    - One way to give this a frequentist interpretation is to consider all the channels that might have been developed by the same process, and then consider the proportion of those that are imperfect
- Note that we are assuming p_A and p_B are known
- What is the probability of system failure?

Aleatory Uncertainty for 1oo2 Architectures

$$
\begin{aligned}
&P(\text{system fails [on randomly selected demand]} \mid pfd_A = p_A, pnp_B = p_B)\\
&= \quad P(\text{system fails} \mid A \text{ fails}, B \text{ imperfect}, pfd_A = p_A, pnp_B = p_B)\\
&\qquad\quad \times P(A \text{ fails}, B \text{ imperfect} \mid pfd_A = p_A, pnp_B = p_B)\\
&\qquad + P(\text{system fails} \mid A \text{ succeeds}, B \text{ imperfect}, pfd_A = p_A, pnp_B = p_B)\\
&\qquad\quad \times P(A \text{ succeeds}, B \text{ imperfect} \mid pfd_A = p_A, pnp_B = p_B)\\
&\qquad + P(\text{system fails} \mid A \text{ fails}, B \text{ perfect}, pfd_A = p_A, pnp_B = p_B)\\
&\qquad\quad \times P(A \text{ fails}, B \text{ perfect} \mid pfd_A = p_A, pnp_B = p_B)\\
&\qquad + P(\text{system fails} \mid A \text{ succeeds}, B \text{ perfect}, pfd_A = p_A, pnp_B = p_B)\\
&\qquad\quad \times P(A \text{ succeeds}, B \text{ perfect} \mid pfd_A = p_A, pnp_B = p_B)
\end{aligned}
$$

Assume, conservatively, that if A fails and B is imperfect, then B will fail on the same demand

$$
\leq \quad 1 \times P(A \text{ fails}, B \text{ imperfect} \mid pfd_A = p_A, pnp_B = p_B) + 0 + 0 + 0
$$

Aleatory Uncertainty for 1oo2 Architectures (ctd.)

$$
\begin{aligned}
& P(A\,\mathrm{fails}, B\,\mathrm{imperfect} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\
= \quad & P(A\,\mathrm{fails} \mid B\,\mathrm{imperfect}, \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\
& \quad \times P(B\,\mathrm{imperfect} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B)
\end{aligned}
$$

(Im)perfection of B tells us nothing about the failure of A on this demand; hence,

$$
\begin{aligned}
= \quad & P(A\,\mathrm{fails} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\
& \quad \times P(B\,\mathrm{imperfect} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\
= \quad & p_A \times p_B
\end{aligned}
$$

Compare with two (un)reliable channels, where failure of B on this demand does increase likelihood A will fail on same demand

$$
\begin{aligned}
& P(A\,\mathrm{fails} \mid B\,\mathrm{fails}, \mathit{pfd}_A = p_A, \mathit{pfd}_B = p_B) \\
\geq \quad & P(A\,\mathrm{fails} \mid \mathit{pfd}_A = p_A, \mathit{pfd}_B = p_B)
\end{aligned}
$$

Aleatory Uncertainty for 1oo2 Architectures (ctd. 2)

I could have factored the conditional probability involving the perfect channel the other way around:

$$\begin{aligned} & P(A\,\text{fails}, B\,\text{imperfect} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\ = \quad & P(B\,\text{imperfect} \mid A\,\text{fails}, \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\ & \times P(A\,\text{fails} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \end{aligned}$$

You might say knowledge that A has failed should affect my estimate of B's imperfection, but we are dealing with aleatory uncertainty where these probabilities are known; hence

$$\begin{aligned} = \quad & P(B\,\text{imperfect} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\ & \times P(A\,\text{fails} \mid \mathit{pfd}_A = p_A, \mathit{pnp}_B = p_B) \\ = \quad & p_B \times p_A \text{ as before} \end{aligned}$$

Note: the claim must be perfection, other global properties (e.g., proven correct) are not aleatory (they are reducible)

Epistemic Uncertainty for 1oo2 Architectures

- We have shown that the events "A fails" "B is imperfect" are conditionally independent at the aleatory level
- Knowing aleatory probabilities of these allows probability of system failure to be conservatively bounded by $p_A \times p_B$
- But we do not know p_A and p_B with certainty: assessor formulates beliefs about these as subjective probabilities
- The beliefs may not be independent, so they will be represented by a joint probability density function $dF(p_A, p_B) = P(pfd_A < p_A, pnp_B < p_B)$
- The unconditional probability of system failure is then

$$P(\text{system fails on randomly selected demand})$$

$$= \int_{\substack{0 \le p_A \le 1 \\ 0 \le p_B \le 1}} p_A \times p_B \, dF(p_A, p_B)$$

(That's a Riemann-Stieltjes integral)

Reliability Estimate for 1oo2 Architectures

- The only source of dependence is in the assessor's bivariate density function $dF(p_A, p_B)$
- But it is really hard to elicit such bivariate beliefs
- What stops beliefs about the two parameters being independent?
- It's not difficulty variation over the demand space
    - Formal verification is uniformly credible
- Surely, it's concern about common-cause errors such as misunderstood requirements, common mechanisms, etc.
- So combine all beliefs about common-cause faults in a third parameter C
    - Place probability mass C at point $(1, 1)$ in (p_A, p_B)-plane as subjective probability for such common faults

Reliability Estimate for 1oo2 Architectures (ctd.)

- With probability C, A will fail with certainty, and B will be imperfect with certainty (and conservatively assumed to fail)
- If assessor believes all dependence between his beliefs about the model parameters has been captured conservatively in C, the conditional distribution factorizes, so

$$P(\text{system fails on randomly selected demand})$$

$$= \quad C + (1 - C) \times \int_{0 \le p_A < 1} p_A \, dF(p_A) \times \int_{0 \le p_B < 1} p_B \, dF(p_B)$$

$$= \quad C + (1 - C) \times P_A^* \times P_B^*$$

where P_A^* and P_B^* are the means of the marginal distributions excluding $(1, 1)$

Reliability Estimate for 1oo2 Architectures (ctd. 2)

- If C is small (as will be likely), can approximate as

$$C + P_A \times P_B$$

  where P_A and P_B are the means of the marginal distributions

- Construct probability C by considering top-level development
    - Or by claim limits (10^{-5})
- Construct probability P_A by statistically valid random testing (10^{-3})
- Construct probability P_B by considering mechanically checked formal verification (see later) (10^{-3})
- Hence overall system *pfd* is about 1.1×10^{-5}

Failures of Commission

- Focus so far is failure of omission
  - e.g., not shutting down reactor when you should
- Also need to consider failures of commission
  - i.e., shutting down reactor when you should not
  - Failure of either channel can do this
- Failures of commission can be mere nuisances, have economic cost, or be safety-critical
- Have to be careful about demands (points in time) vs. nondemands (absence of demands over intervals of time)
- Discretize time: e.g., single flight of an aircraft
- Can then use pfds for both demands and nondemands

Failures of Commission

- By similar arguments as before, get

  $$P(\text{system fails on randomly selected nondemand} \,|\, pfd_A = p_{A2}, pnp_B = p_{B2})$$

  $$= \quad p_{A2} + p_{B2} - p_{A2} \times p_{B2}$$

- where p_{A2} and p_{B2} are aleatory probabilities of failure and imperfection, respectively, for A and B wrt. failures of commission
- This result shows us that the diversity in a 1oo2 architectures provides no benefit with respect to these failures
- For epistemic assessment, conservative to ignore final term, do not then need a factoring argument for epistemic values
- So system *pfd* wrt. failures of commission is $P_{A2} + P_{B2}$ where P_{A2} and P_{B2} are means of the marginal distributions

Risk of Failures

- Denote the consequence (cost) of a failure of omission by c_1, and the consequences of failures of commission by the A and B channels by c_{A2} and c_{B2}, respectively
    - The costs are different because the two channels may operate in different ways
- Denote the probability that a randomly selected interval triggers a demand by f
- Then epistemic risk is bounded by

$$f \times c_1 \times (C + P_{A1} \times P_{B1}) + (1 - f) \times c_{A2} \times P_{A2} + (1 - f) \times c_{B2} \times P_{B2}$$

omission + commission

Assurance Case for Formal Verification

- How might we construct probabilities $P_{B1}, P_{B2} \leq 10^{-3}$?
- i.e., less than 1 in 1,000 chance that the monitor is imperfect
- We will formally verify or formally synthesize the monitor
  - i.e., prove it correct using automated tools
- What are the dominant hazards to this process?
  - Topics outside formal analysis (e.g., compiler bugs)—those have to be included in C
    - ⋆ Can be verified by testing (autogenerated from specs)
  - Incorrect claims—that's dealt with in C, too
  - Incorrect formalization of claims and supporting theories
  - Unsound formalization of these (e.g., flawed axioms)
  - Unsound theorem prover or monitor synthesis

Soundness Guarantees for Formal Verification

- Unsound axiomatizations can be eliminated by constructive methods, or by exhibiting a constructive model
- Of the remaining hazards, incorrect formalization of the claims and theories are surely dominant
    - Allocate most of our 10^{-3} "budget" here
- Then, an adequate soundness guarantee for our theorem prover or formal synthesis procedure will be about 10^{-4}
- This is not a very demanding requirement

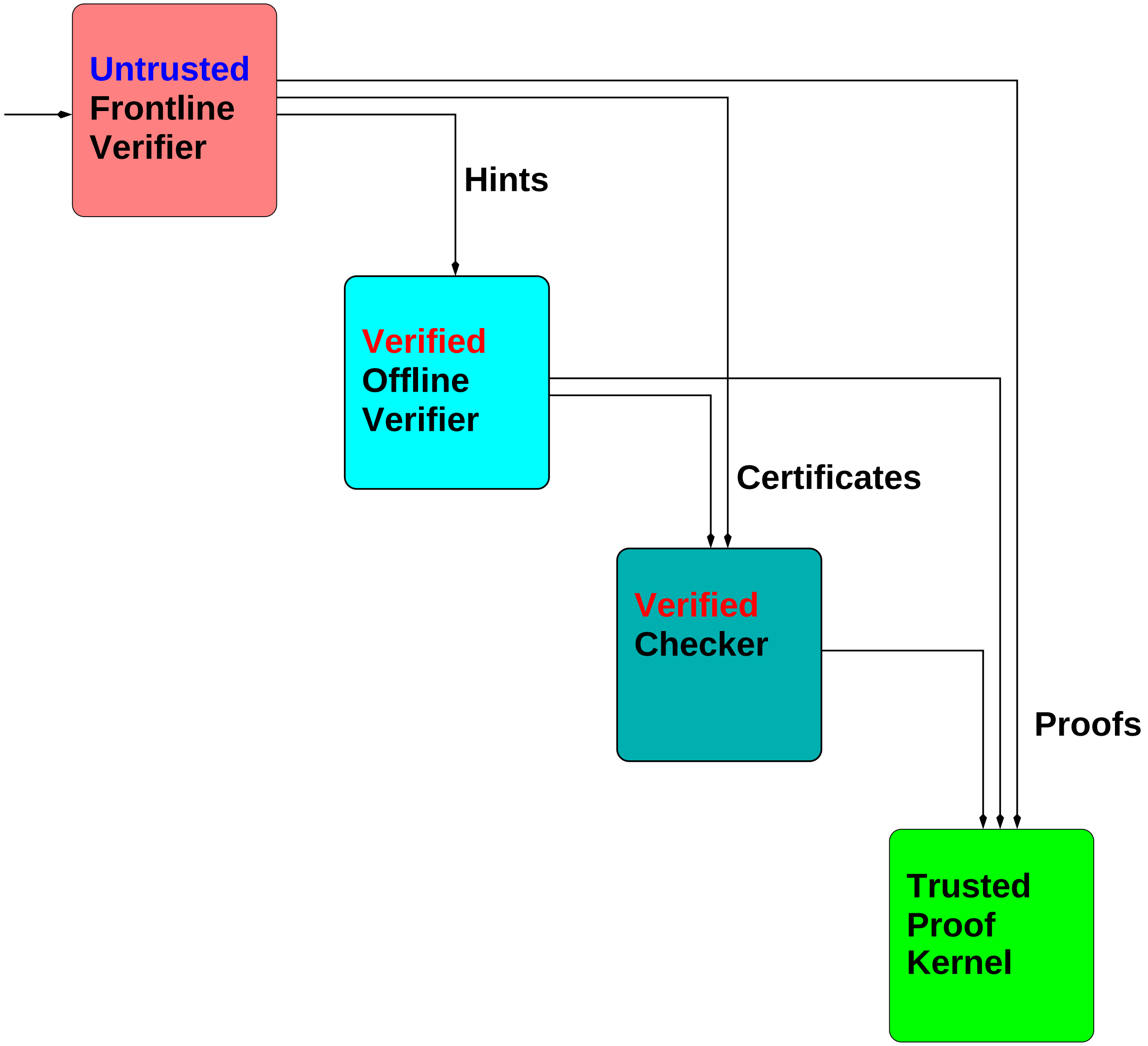
Soundness Guarantees for Formal Verification Tools

- A verification will certainly fail if your tools and deductive components lack the power to complete it
- We need ways to guarantee soundness that do not compromise deductive power
- Many options: computational reflection, diverse verifiers, trusted core, proof generation and verified checker
- Computational reflection is fine, but has to build on something more basic
- Diversity has well-known weaknesses
- Trusted core is slow, and a weak guarantee
  - Even the relatively solid and small (~ 400 lines of OCaml) HOL Light core was found to have two soundness bugs.
  - Has since been (self) verified

Proof Generation and Verified Checkers

- Traditional approach is to generate primitive proof objects that can be independently checked by a verified proof kernel
  - An instance of an operational system with a monitor!
- Problem is the primitive proof objects from powerful provers (e.g., SMT solvers) are vast (gigabytes)
- We favor more powerful checkers and offline verifiers that can be driven by more succinct certificates and hints, respectively
  - Developing and formally verifying useful checkers and offline verifiers is a major research challenge
  - A high-performance SAT solver is a good start: checking of many verifiers can be reduced to SAT plus something
  - Shankar and Marc Vaucher have verified a modern SAT solver in PVS; the formal specification is efficiently executable (modulo lacunae in the PVS evaluator)

Verified Reference Kernels



Untrusted Frontline Verifier

Hints

Verified Offline Verifier

Certificates

Verified Checker

Proofs

Trusted Proof Kernel

Software IVHM for Aircraft

- Requirements for safety critical software in aircraft are extreme (e.g., probability of failure 10^{-9}/hour)
- Retrospective evidence it was achieved
  - At least, until recent accidents and incidents
  - A330 accident near Perth, 777 incident near Perth, A340 incident near Schiphol, 737 crash at Schiphol
- But how to assess it prospectively, in certification?
- Skepticism it can be achieved by analysis alone
  - e.g., CAST 24 report: suggests diversity
- IVHM is Integrated Vehicle Health Maintenance
  - Monitoring, prognosis, mitigation etc.
- Software IVHM applies this to software

A Recent Incident Due to Software

- An Airbus A340 en-route from Hong Kong to London on 8 February 2005
- Toward the end of the flight, two engines flamed out, crew found certain tanks were critically low on fuel, declared an emergency, landed at Amsterdam
- Two Fuel Control Monitoring Computers (FCMCs) on this type of airplane; they cross-compare and the "healthiest" one drives the outputs to the data bus
- Both FCMCs had fault indications, and one of them was unable to drive the data bus
- Unfortunately, this one was judged the healthiest and was given control of the bus even though it could not exercise it
- Further backup systems were not invoked because the FCMCs indicated they were not both failed

Software Health Management and Monitoring

- System hazards due to software faults are a topic of concern in aviation safety: one accident, and several serious incidents
- Traditional approach is fault avoidance
  - Strive to eliminate software faults
  - The intent of DO-178B, DO-297, etc.

  May be reaching the limits of effectiveness
- So consider buttressing it by software health management
  - Techniques for monitoring, diagnosing, prognosing, and mitigating the manifestations of residual faults.
- But what specifications do we monitor against?
  - DO-178B does a good job ensuring the software correctly implements its low and high level specifications
  - Faults are likely to be in these specifications

  Need higher-level, independent specifications

Safety Cases and Formal Monitors

- Intellectual basis for assurance in support of certification is a credible argument based on documented evidence that supports suitable claims
- DO-178B is an example of standards-based assurance
  - Specifies just the evidence to be developed
  - The claims and argument are largely implicit

  Effective in slow-moving fields, but can be a barrier and a hazard to innovation
- Hence, growing interest in safety-case approach to assurance
  - Make all of the argument, claims, evidence explicit
- Aha: monitor against the (sub)claims in the safety case
- Formal monitors are synthesized from or verified against safety claims using automated formal methods

Interpretation for Formal Monitors

- In a monitored architecture
  - Have an operational channel A completely responsible for functions of the system
  - And a monitor B that can trigger an alarm if it sees violation of safety properties
  - Requires higher level fault-recovery
  - So really an subsystem architecture
- Reuse previous analysis, where A has only failures of omission
- Demands arrive at some constant rate per unit time
- Nondemands arrive each time A succeeds
- Hence,

$$\text{risk/unit time} \le c_1 \times (C + P_{A1} \times P_{B1}) + (1 - P_{A1}) \times c_2 \times P_{B2}$$

Consequences For Formal Monitors

- Our analysis yields prob. of failure wrt. failures of omission in monitored system as $(C + P_{A1} \times P_{B1})$, vs. P_{A1} without monitor
- Credible and modest claims for perfection of a monitor (e.g., $P_{B1} < 10^{-3}$) deliver useful improvement
- Provided probability of common cause faults C is small
- I think it can be, because the monitor is derived from the safety case

Consequences For Formal Monitors (ctd.)

- But we also need to be concerned about failures of commission: risk is $c_2 \times P_{B2}$
- These depend on the monitor alone
- Cost of these failures must be commensurate with credible claims for probability of perfection
  - A340 fuel system monitor: warn pilot—OK
  - A300 roll rate anomaly: reboot EFIS bus—not OK
- Imperfection wrt. failures of commission likely depends more on selection of monitored properties than correctness of the monitor
- Hence, selection of these properties is critical

Summary

- Started with analysis of 1oo2 systems
    - Failure of one channel and imperfection of the other are conditionally independent at the aleatory level
    - Only dependence is in epistemic assessment of their probabilities
    - Dependencies can be absorbed in a common-cause probability C
- The analysis was extended to failures of commission
- Then carried over to monitored systems
- And the epistemic failure rates and risk depend on $C,\ P_{A1},\ P_{B1},\ P_{B2}$ and $f,\ c_1,\ c_{B2}$
- It is feasible to assess these parameters

Conclusions

- Asymmetric 1oo2 systems, and monitored systems are plausible ways to achieve high reliability
- With a possibly perfect channel they also provide a credible way to assess it
- Risk of failures of commission (false alarms) requires careful consideration and engineering: for formal monitors, focus should be on choice of monitored properties
- Reasonable rates of perfection require only modest guarantees for the prover; suggested how these can be provided without compromising performance
- Caution: focus was on failure of monitored subsystems—we still have to respond to those failures at the system level

Research Topics

- Can significant properties be monitored at the subsystem level, or are they emergent?
- More generally, can we develop approaches to assurance cases that are compositional?
  - Given the cases for components
  - Assemble these to provide case for system
  - Or for new context of deployment

  These are very difficult topics (cf. IMA)
- We have a plausible approach for NSA-grade security
  - The MILS approach
- Yet more generally, can we assess assurance cases reliably?
  - Currently, it's all human judgement
  - Reserve this for where it's really indispensable
  - Formalize and automate all that can be