Assurance for Complex Systems

John Rushby and N. Shankar

Computer Science Laboratory
SRI International
Menlo Park, CA

Introduction

- Our research group (Formal Methods and Dependability) at SRI has a long-standing interest in avionics safety assurance.

- We have developed tools such as PVS, SAL, HybridSAL, Yices 1 & 2, PCE (Probabilistic Consistency Engine), and ETB (Evidential Tool Bus).

- Applications within NASA include fault tolerance (RCP, SPIDER, TTA) and air-traffic control (KB3D, AILS).

Credo

- Assurance demonstrates that everything has been anticipated
  - And being sure nothing really bad is in there
- Complex systems mean that there's a lot of everything
- So we need ways to develop assurance compositionally
  - i.e., in a modular fashion, from the assurance of systems to that of systems of systems
- And we need sound, credible, and efficient ways to develop assurance for individual systems
- We're mostly concerned with software (and its interaction with the environment)
  - Because that's where all the complexity is
  - Mostly in redundancy management

Overview

- Standards- vs. argument-based assurance
- Formal methods in argument-based assurance
- Formal monitors
- Compositional approaches to system properties

Standards-Based Assurance

This is current practice—for example:

- ARP 4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment
- ARP 4754: Certification Considerations for Highly-Integrated or Complex Aircraft Systems
- DO-297: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations
- DO-254: Design Assurance Guidelines for Airborne Electronic Hardware
- DO-178B: Software Considerations in Airborne Systems and Equipment Certification

Works well in fields that are stable or change slowly

- Can institutionalize lessons learned, best practice
  - e.g. evolution of DO-178 from A to B to C

But less suitable with novel problems, solutions, methods

A Recent Incident

- Fuel emergency on Airbus A340-642, G-VATL, on 8 February 2005 (AAIB SPECIAL Bulletin S1/2005)
- Toward the end of a flight from Hong Kong to London: 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

Implicit and Explicit Factors

- See also ATSB incident reports for in-flight upsets of Boeing 777, 9M-MRG (Malaysian Airlines) and Airbus A330 VH-QPA (QANTAS), near Perth Australia
- How could gross errors like these pass through rigorous assurance standards?
- Maybe effectiveness of current certification methods depends on implicit factors such as safety culture, conservatism
- Current business models are leading to a loss of these
  - Outsourcing, COTS, complacency, innovation
- Surely, a credible certification regime should be effective on the basis of its explicit practices
- How else can we cope with challenges of more complex systems?

Standards and Argument-Based Assurance

- All assurance is based on **arguments** that purport to justify certain **claims**, based on documented **evidence**
- Standards usually define only the evidence to be produced
- The claims and arguments are implicit
- Hence, hard to tell whether given evidence meets the intent
- E.g., is MC/DC coverage evidence for good testing or good requirements?
- Recently, argument-based assurance methods have been gaining favor: these make the elements explicit

The Argument-Based Approach to Software Certification

- E.g., UK air traffic management (CAP670 SW01), UK defence (DefStan 00-56), growing interest elsewhere
- Applicant develops a safety case
  - Whose outline form may be specified by standards or regulation (e.g., 00-56)
  - Makes an explicit set of goals or claims
  - Provides supporting evidence for the claims
  - And arguments that link the evidence to the claims
    - ⋆ Make clear the underlying assumptions and judgments
    - ⋆ Should allow different viewpoints and levels of detail
- Generalized to security, dependability, assurance cases
- The case is evaluated by independent assessors
  - Explicit claims, evidence, argument

Formal Methods In Argument-Based Assurance

- Standards-based methods at least establish a floor
- But how do you know if an argument-based case is really sound?
  - A lot of expert judgement
  - But the main argument ought to follow by enumeration of assumptions, modeling of designs, and standard laws of reasoning
- This is what formal methods do, and have the advantage over simulation and testing that they consider all cases
- Because they do the analysis for symbolic values x, y, z, rather than explicit numbers
- Highly automated in modern methods (e.g., Simulink Design Verifier)

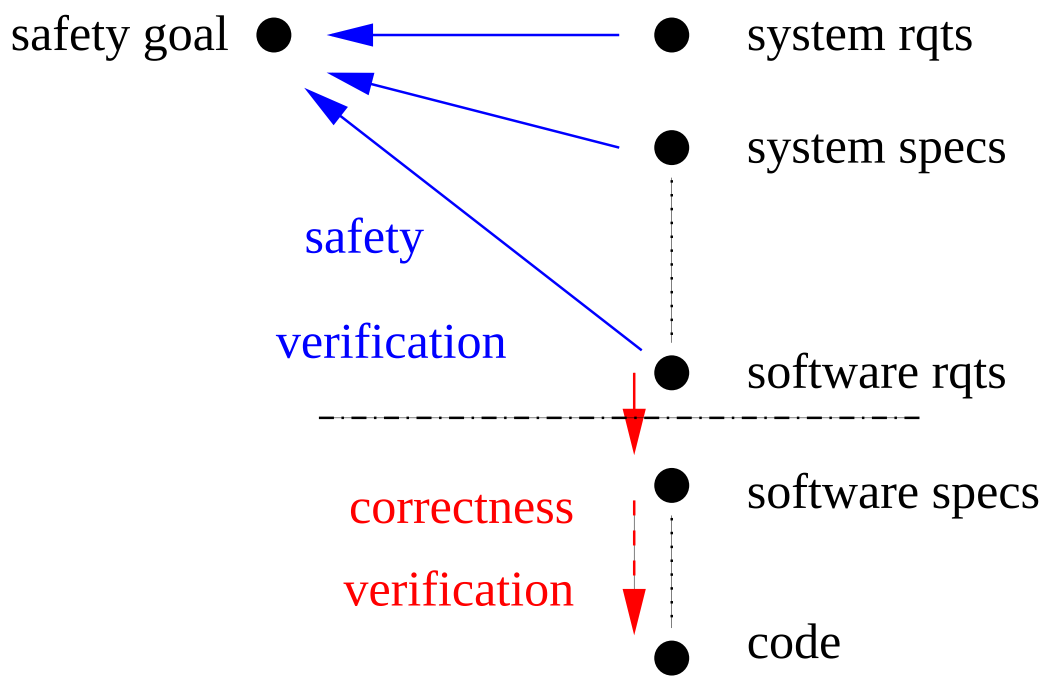
Formal Methods In Argument-Based Assurance (ctd.)

- A lot of safety assurance is about enumerating hazards/threats and showing these are countered effectively
  - FTA, FMEA, HAZOP are ways to enumerate hazards
- How do we know we have considered all hazards?
- Formal methods force complete enumeration of the the assumptions $A_1, \ldots, A_n$ under which the system S satisfies the requirements R

$$A_1, \ldots, A_n, S \vdash R$$

- Can then do safety analysis on each assumption A_i
  - i.e., ask what if it is false?
  - and how could it be falsified?
- We are exploring formal mechanization of these

Software Standards Focus on Correctness Rather than Safety



- Premature focus on correctness is hugely expensive
  argument-based methods could reduce this
- Can also allow runtime checking of safety properties

Formal Monitors

- An attractive idea is to monitor software systems for violation of safety requirements
- Trigger higher-level fault management when violations detected
- Does no good to monitor against software requirements
    - DO-178B guarantees these are implemented correctly
    - The problems are always in the software requirements
- So monitor against the assertions in the safety case
- Formal monitors are synthesized or verified to correctly check those assertions

Formal Monitors (ctd.)

- Monitoring is a form of diverse redundancy
- Its known that the reliability of diverse systems cannot be deduced by multiplying the reliabilities of the individual channels
  - There will be correlated failures
- Rather than consider the reliability of a formal monitor, consider its possible perfection
- Then has a probability of imperfection
- But Littlewood and Rushby show that the failure of an operational channel and the imperfection of a monitor are independent at the aleatory level
  - Argument that it extends to the epistemic level
- Hence, can multiply these probabilities: a .999 operational channel a .999 monitor give you a .999999 system

Systems and Components

- The FAA certifies airplanes, engines and propellers
- Components are certified only as part of an airplane or engine
- That's because it's the interactions that matter and it's not known how to certify these compositionally
- But modern engineering and business practices use massive subcontracting and component-based development that provide little visibility into subsystem designs
- Furthermore, the binding times for system architectures and for component behaviors are being delayed
- And adaptive systems may have undesired emergent behavior due to interactions
- So we are forced to contemplate compositional and incremental approaches to certification

Compositional and Incremental Certification

- These are immensely difficult
  - The assurance case may not decompose along architectural lines

  Profound insight (Ibrahim Habli & Tim Kelly)
- But, in some application areas we can insist that it does
  - Goes to the heart of what is an architecture
- A good one supports and enforces the safety case
- Interactions use only known, intended mechanisms
  - No unprotected IPC channels
  - No signaling through cache occupancy, etc.
  - No unmodeled interaction through the controlled plant
- This is what partitioning in IMA is all about
- And the MILS approach to security

Related Projects

- Certification of SRI's M7 telesurgery robot
- *Cybertrails* reactive analysis of audit trails
- Verified Reference Kernel for checking formal claims

Closing Thoughts And Questions

- What is the right approach for developing and certifying safe software-based systems?
- And are safety cases with explicit evidence the way to go?
- Do formal monitors deliver greater assurance?
- How do we move toward explicitly compositional certification?