Formal Techniques for Software IVHM

Natarajan Shankar & John Rushby (SRI International)

Objectives

- Develop a framework for *assurance cases* for aviation software safety based on *explicit evidence*
- Complement *probably reliable* software with *possibly perfect* monitors that detect, diagnose, and mitigate in-flight software anomalies
- Achieve possible perfection through *formally verified monitors* (FVMs)
- Certify software-based systems to high levels of reliability

Technical Challenges

- Software does not have a well-defined factor of safety
- Reliability cannot be accurately estimated from test data
- Replication does not increase confidence
- Requirements can be incorrect or incomplete
- Software anomalies might not be observable at subsystem level
- Safety mechanisms must detect all anomalies with few false alarms
- Software verification is hard
- No clear connection between correctness and reliability
- Assurance case is not well-specified

Technical Approach

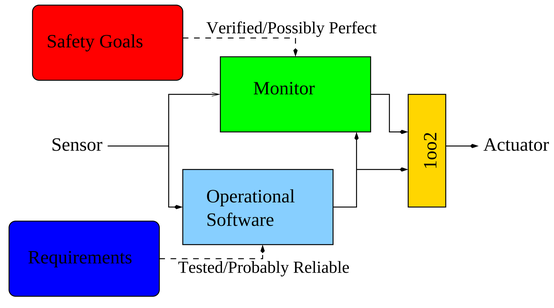
- Identify the *safety case* for the system
- Develop *monitors* for the software behavior
- The monitors are *formally verified* against or *synthesized from* the safety case – monitors are relatively simple and stable
- Failure on input of the monitor is conditionally independent to that of the operational channel.
- Probability of failure (of omission) on demand is multiplicative
- Analysis accounts for *aleatory* probability and *epistemic* uncertainty

Analysis

- Several recent accidents and incidents are due to software
  - 8-1-2005: In-flight upset of 9M-MRG B777: fusion/fault management in ADIRU
  - 2-8-2005: Fuel emergency on G-VATL A340: fault management in fuel control subsystem
  - 10-7-2008: Violent pitching of VH-QP A330: fusion/fault management in AOA sensors
- Software anomalies involve interaction between physical and virtual components
- Fusion and fault management are common sources of failure
- Requirements can be flawed, so replication and diversity do not increase reliability
- Requisite levels of assurance cannot be obtained by testing

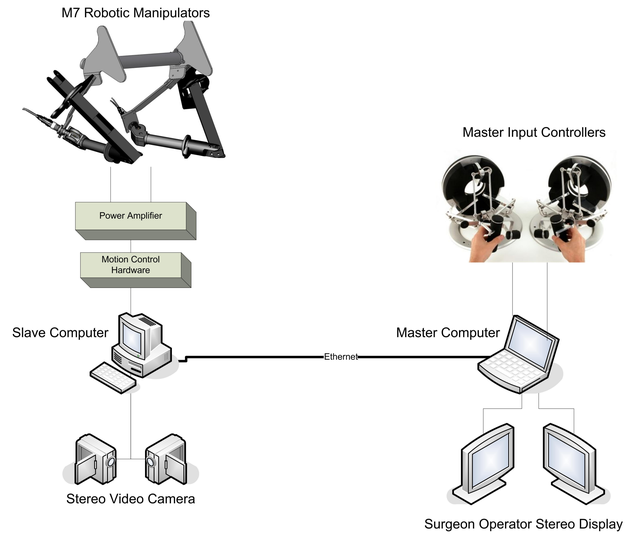
Solution

- Safety goals are simpler than functional requirements
- Violation of safety goals can often be detected
- Safety monitors are simple enough to be verifiable
- Plausible claim for possible perfection (independent of input) of monitors
- Software and monitor fail independently $\Longrightarrow$ possible perfect increases reliability; simplifies assessment



Results and Publications

- Rigorous reliability analysis for 1oo2 systems combining a probably reliable operational channel with a formally verified (possibly perfect) monitor
- Type-based verification and test-case generation for formal monitors in Simulink
- Validation on SRI's safety-critical M7 robotic telesurgery system



A Safety-Case Approach For Certifying Adaptive Systems, by John Rushby. AIAA Infotech@Aerospace Conference, April 2009.

Reliability Of Diverse Two-Channel Systems In which One Channel is "Possibly Perfect", by Bev Littlewood and John Rushby. Technical Report SRI-CSL-09-02, May 2009

Software Verification and System Assurance, by John Rushby. Invited paper at Software Engineering and Formal Methods, November 2009.

Conclusions

- Software reliability is hard to estimate
- Impact of software correctness on reliability is not clear
- Possible perfection is the bridge between correctness and reliability
- Evidence-based assurance cases
- Simulink-based verification framework for safety monitors
- Validation on M7 robot testbed

Sponsored by the Integrated Vehicle Health Management (IVHM) Project