Safety, Ethics, and Professionalism

55:036
Embedded Systems and System Software

Safety, Ethics, Professionalism

- Embedded systems are being increasingly used in critical applications
  - safety-critical
    - potential for death or injury
    - loss or damage to property
  - societally-critical
    - potential for major disruption to everyday life
  - business-critical
    - potential for large economic loss to company/business
    - exposure to litigation

Role/Responsibility of Embedded System Engineer

- Act professionally and ethically
- Understand and appreciate risk factors
- insure that proper attention is paid to these factors in system design and implementation
  - interlocks
  - cross-checks
  - independent monitoring of critical functions
  - etc.
- Insure that thorough analysis and testing is done
  - long-term
  - anomalous situations

Ethical Responsibilities

- Errors of Commission
  - falsifying test results
  - covering up known or suspected problems
  - knowingly leaving potentially serious bugs in a product
  - knowingly violating safety/reliability standards
- Errors of Omission
  - failure to adequately analyze potential failure modes and consequences
  - failure to adequately test a system
  - failure to design and implement appropriate safety/robustness features
  - failure to react aggressively enough to reports/warnings of possible problems
  - others???

Three Quick Case Studies

- Therac-25 Radiation Therapy Accidents
- Ariane 5 Rocket Failure
- Patriot Missile System Failure

The Therac-25 Accidents

- Computerized Radiation Therapy Machine
- Designed in early 1980s
- Resulted in Six Major Accidents between June 1985 and January 1987
  - At least two deaths
  - Several serious injuries

Therac-25 Overview

- Linear Particle Accelerator
- Replaced earlier version
- Utilized much more computerized control
- In particular, more software responsibility for safety maintenance
- Reused some software from earlier versions.
- Fault analysis considered only computer hardware failures

Therac-25 Accident History

- First accidental overdoses reported in 1985
- Manufacturer could not reproduce accident scenarios
- Suspected hardware (microswitch) problems and did redesign
- Did not include independent interlock to prevent overdose

Therac-25 Accident History--continued

- Accidents continued in 1986 and 87
- traced to operator behavior (keyboard entry)
  - timing related
- Several different software problems eventually implicated
  - related to concurrency
  - lack of locking/atomic operations for access to shared variables

Therac-25 Retrospective

- Embedded software was designed and implemented by one engineer
- Fairly stringent real-time constraints
  - preemptive schedule
  - 100 msec. scheduling granularity
- No real synchronization of access to shared variables
- Many potential race conditions, with relatively low probability of occurrence
- Most accidents ultimately traced to synchronization problems (race conditions) in processing operator input from the keyboard
  - resulted in improper settings
  - no independent feedback to warn operator

Therac-25 Retrospective (Continued)

- Ethical Issues:
  - The vendor appears, at the very least, to be guilty of:
    - inadequate software-engineering practices
    - inadequate/flawed fault analysis
    - failure to implement needed cross-check and/or interlock features to prevent accidental overdose
    - failure to react aggressively to initial reports of problems

Ariane 5 Rocket Failure

- First test launch of French Ariane 5 rocket
- June, 1996
- Self-destructed due within 40 seconds after lift-off due to software anomaly

Ariane 5 Rocket Failure--Cause

- Floating point exception generated after lift-off by a software module that was concerned only with missile/launchpad alignment prior to launch
- Alignment module was reused from earlier guidance system
- Module remained operational for 50 seconds after launch
  - This had been a requirement in the earlier application
- Testing procedures had not considered the behavior of the alignment module after lift-off
- Interestingly, the rocket guidance system was completely replicated to protect against hardware failures, but this was of no use for this failure condition

Patriot Missile System Failure

- Patriot Missile Defense System used for first time in first Gulf War (1991)
- Failed to intercept incoming Scud missile
- Missile struck U.S. Army Barracks, killing 28 soldiers and wounding over 100
- GAO investigation ultimately implicated software

The Patriot Missile Software Problem

- System's internal clock measured time in 100 msec. units
- Multiplied by 1/10 to convert to seconds
- Calculation performed using a 24 bit register.

The Problem

- $1/10 = 1/2^4 + 1/2^5 + 1/2^8 + 1/2^9 + 1/2^{12} + 1/2^{13} + \ldots$
- Truncated to 24 bits to fit into register
- Resulted in cumulative timing error
- For each calculation this error was very small:
  - approx. 0.000000095
- However, after 100 hours of operation, this error was approximately 0.34 seconds
- A Scud missile travels more than 1 kilometer in this time
- Error resulted in Scud being outside of the Patriot's "range gate"
- Interestingly, this error had been corrected in some parts of the Patriot's software but not in others. This is what ultimately caused the failure

What Can We Learn From These Case Studies

- Note that the specific causes were quite different in each failure situation
- All systems underwent stringent design review and testing
- So, what went wrong?

Case Studies—What Went Wrong?

Hopefully not this!!

- Errors of Commission
  - falsifying test results
  - covering up known or suspected problems
  - knowingly leaving potentially serious bugs in a product
  - knowingly violating safety/reliability standards
- Errors of Omission
  - failure to adequately analyze potential failure modes and consequences
  - failure to adequately test a system
  - failure to design and implement appropriate safety/robustness features
  - failure to react aggressively enough to reports/warnings of possible problems
  - others???

Case Studies—What Went Wrong?

- Errors of Commission
  - falsifying test results
  - covering up known or suspected problems
  - knowingly leaving potentially serious bugs in a product
  - knowingly violating safety/reliability standards
- **Errors of Omission**
  - **failure to adequately analyze potential failure modes and consequences**
  - **failure to adequately test a system**
  - **failure to design and implement appropriate safety/robustness features**
  - **failure to react aggressively enough to reports/warnings of possible problems**
  - **others???**

What Does All Of This Have To Do with Ethics and Professionalism?

- Ethics-Personal code of behavior
  - responsibility
  - accountability
  - determination of right vs. wrong
- Engineers are often in the best position to determine the ethical consequences of their actions (or inactions) or those of others with whom they interact.

Examples of Ethical Issues

- Have sufficient safety/robustness features been designed into this system?
- Has it been adequately validated and tested?
- Have corners been cut?
- Has engineering integrity been compromised by cost or marketing considerations?
- Have problems been covered up?

Special Ethical Issues for Safety Critical Systems

- How much is a life worth?
- What is the proper balance between product safety and cost?
  - Would you allow yourself to be pressured into removing a safety feature if you were convinced that this could eventually result in death or injury?
- Are any systems really non-safety critical?
- What about other forms of criticality: societal, economic?

Professional Ethics

- Professional Societies have ethics codes
  - IEEE
  - ACM
  - etc