Embedded Software Design

Case Studies

Pao-Ann Hsiung

National Chung Cheng University

Contents

- Case Studies
  - Therac-25
  - Ariane 501
  - USS Yorktown
  - Patriot

Therac-25

1985 ~ 1987

AECL Development History

- Therac-6: 6 MeV device,
  - Produced in early 1970's
  - Designed with substantial hardware safety systems and minimal software control
  - Long history of safe use in radiation therapy
- Therac-20: 20 MeV dual-mode device
  - Derived from Therac-6 with minimal hardware changes, enhanced software control
- Therac-25: 25 MeV dual-mode device
  - Redesigned hardware to incorporate significant software control, extended Therac-6 software

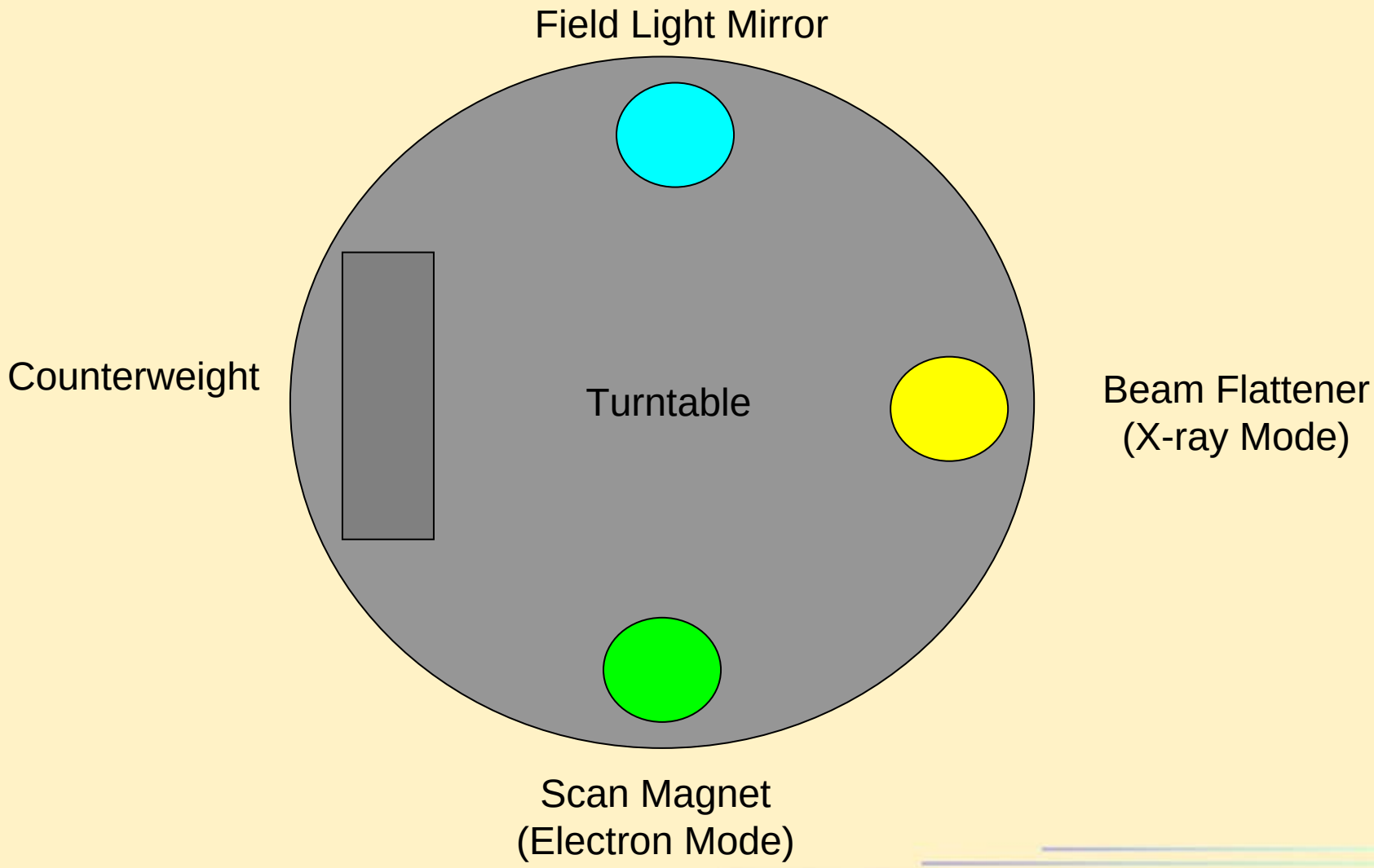
Therac-25

- Medical linear accelerator
  - Used to zap tumors with high energy beams.
    - Electron beams for shallow tissue or x-ray photons for deeper tissue.
- Eleven Therac-25s were installed:
  - Six in Canada
  - Five in the United States
- Developed by Atomic Energy Commission Limited (AECL).

Therac-25

- Improvements over Therac-20:
  - Uses new “double pass” technique to accelerate electrons.
  - Machine itself takes up less space.
- Other differences from the Therac-20:
  - Software now coupled to the rest of the system and responsible for safety checks.
    - Hardware safety interlocks removed.
  - “Easier to use.”

Therac-25 Turntable



Accident History

- June 1985, Overdose (shoulder, arm damaged)
  - Technician informed overdose is impossible
- July 1985, Overdose (hip destroyed)
  - AECL identifies possible position sensor fault
- Dec 1985, Overdose (burns)
- March 1986, Overdose (fatality)
  - “Malfunction 54”
  - Sensor reads underdosage
  - AECL finds no electrical faults, claims no previous incidents

Accident History (cont.)

- April 1986, Overdose (fatality)
  - Hospital staff identify race condition
  - FDA, CHPB begin inquiries
- January 1987, Overdose (burns)
  - FDA, CHPB recall device
- July 1987, Equipment repairs Approved
- November 1988, Final Safety Report

What Happened?

- Six patients were delivered severe overdoses of radiation between 1985 and 1987.
  - Four of these patients died.
- Why?
  - The turntable was in the wrong position.
  - Patients were receiving x-rays without beam-scattering (光散射).

What would cause that to happen?

- Race conditions.
  - Several different race condition bugs.
- Overflow error.
  - The turntable position was not checked every 256th time the “Class3” variable is incremented.
- No hardware safety interlocks.
- Wrong information on the console.
- Non-descriptive error messages.
  - “Malfunction 54”
  - “H-tilt”
- User-override-able error modes.

Cost of the Bug

- To users (patients):
  - Four deaths, two other serious injuries.
- To developers (AECL):
  - One lawsuit
    - Settled out of court
  - Time/money to investigate and fix the bugs
- To product owners (11 hospitals):
  - System downtime

Source of the Bug

- Incompetent engineering.
  - Design
  - Troubleshooting
- Virtually no testing of the software.
  - The safety analysis excluded the software!
  - No usability testing.

Bug Classifications

- Classification(s)
  - Race Condition (System Level bug)
  - Overflow error
  - User Interface
- Were the bugs related?
  - No.

Testing That Would Have Found These Bugs…

- Design Review
- System level testing
- Usability Testing

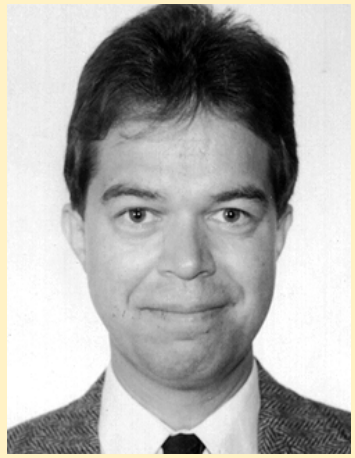
- Cost of testing… worth it?
  - Yes. It was irresponsible and unethical to not thoroughly test this system.

Sources

- **Leveson, N., Turner, C. S., An Investigation of the Therac-25 Accidents.** *IEEE Computer*, Vol. 26, No. 7, July 1993, pp. 18-41. http://courses.cs.vt.edu/~cs3604/lib/Therac_25/Therac_1.html
  - Information for this article was largely obtained from primary sources including official FDA documents and internal memos, lawsuit depositions, letters, and various other sources that are not publicly available.

The authors:

Nancy Leveson



Clark S. Turner

Ariane 501

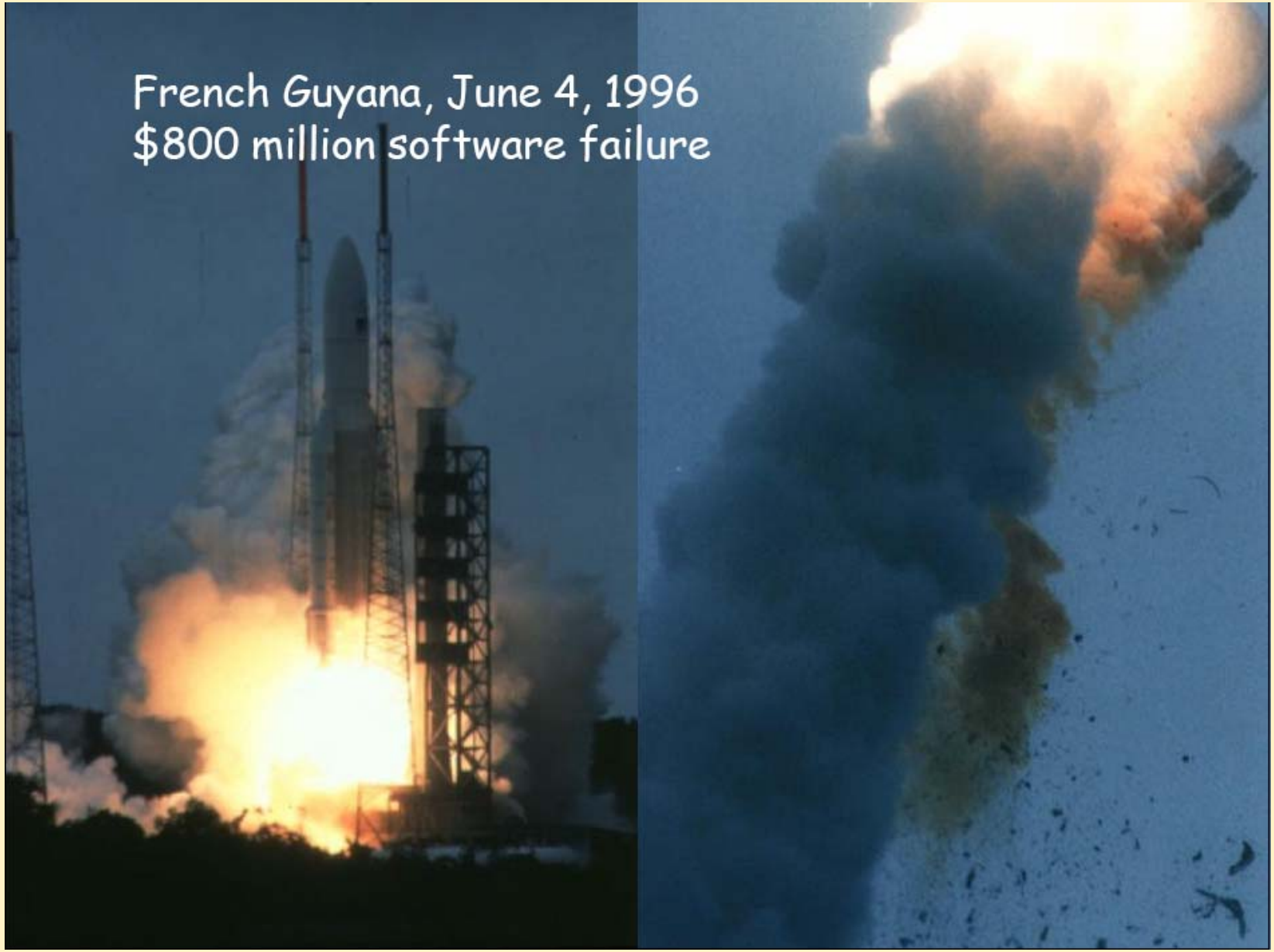
1996

Ariane 501

- On 4 June 1996, the maiden flight of the Ariane 5 launcher ended in a failure.
- Only about **40 seconds** after initiation of the flight sequence, at an altitude of about **3700 m**, the launcher veered off its flight path, broke up and exploded.
- Investigation report by Mr Jean-Marie Luton, ESA Director General and Mr Alain Bensoussan, CNES Chairman
  - ESA-CNES Press Release of 10 June 1996

Ariane 501 Failure Report

- Nominal behavior of the launcher up to H0 + 36 seconds;
- Simultaneous failure of the two inertial reference systems;
- Swivelling into the extreme position of the nozzles (尾噴管) of the two solid boosters (助推器) and, slightly later, of the Vulcain engine, causing the launcher to veer abruptly;
- Self-destruction of the launcher correctly triggered by rupture of the electrical links between the solid boosters and the core stage.


French Guyana, June 4, 1996
$800 million software failure

Sequence of Events on Ariane 501

- At 36.7 seconds after H0 (approx. 30 seconds after lift-off) the computer within the back-up inertial reference system, which was working on stand-by for guidance and attitude control, became inoperative. This was caused by **an internal variable** related to the horizontal velocity of the launcher exceeding a limit which existed in the software of this computer.
- Approx. 0.05 seconds later the active inertial reference system, identical to the back-up system in hardware and software, failed for the same reason. Since the back-up inertial system was already inoperative, correct guidance and attitude information could no longer be obtained and loss of the mission was inevitable.
- As a result of its failure, the active inertial reference system transmitted essentially diagnostic information to the launcher's main computer, where it was interpreted as flight data and used for flight control calculations.

Sequence of Events on Ariane 501

- On the basis of those calculations the main computer commanded the booster nozzles, and somewhat later the main engine nozzle also, to make a large correction for an attitude deviation (偏航) that had not occurred.
- A rapid change of attitude occurred which caused the launcher to disintegrate at 39 seconds after H0 due to aerodynamic forces (空氣動力).
- Destruction was automatically initiated upon disintegration, as designed, at an altitude of 4 km and a distance of 1 km from the launch pad.

Post-Flight Analysis (1/4)

- Inertial reference system of Ariane 5 is same as in Ariane 4
- In Ariane 4
  - Used before launch
  - For realignment of system in case of late hold in countdown
- In Ariane 5
  - No use!!!
  - Retained for commonality reasons
  - Operates for 40 seconds after lift-off
- Horizontal velocity variable
  - Decided not to prevent overflow of values
  - Did not analyze which values would the variable have after lift-off

Post-Flight Analysis (2/4)

- In Ariane 4
  - During first 40 seconds of flight
    - No value overflow possible for the horizontal velocity variable
- In Ariane 5
  - High initial acceleration
  - Horizontal velocity is FIVE times more rapid than Ariane 4
    - Horizontal velocity variable value overflow occurred within 40 seconds!!!

Post-Flight Analysis (3/4)

- In the review process
  - Limitations of alignment software not fully analyzed
  - Possible implications of allowing it to continue to function during flight were not realized
- In the specification and test plans
  - Ariane 5 trajectory data were not included
    - Not tested under simulated Ariane 5 flight conditions
    - Design error was not discovered

Post-Flight Analysis (4/4)

- In overall system simulation
  - Decided to use simulated output of inertial reference system, not the system itself or its detailed simulation
  - Could have included entire inertial reference system in overall system simulation
- In post-flight simulations
  - Software in inertial reference system + actual trajectory of Ariane 501 flight
  - Faithfully reproduced the chain of events leading to the failure of inertial reference systems

USS Yorktown

1998

USS Yorktown

- The Yorktown lost control of its propulsion system because its computers were unable to divide by the number zero, the memo said.
- The Yorktown's Standard Monitoring Control System administrator entered zero into the data field for the Remote Data Base Manager program.
- The ship was completely disabled for several hours

USS Yorktown

- This is such a dumb bug there is little need to comment!
- All input data should be checked for validity
- If you have a zero divide risk then trap it
- Particularly if it might bring down an entire warship
- And, even if a zero divide gets through, how robust is a system where a single user input out of range error can crash an entire ship?

Patriot

1991

Patriot

- On February 25, 1991, during the Gulf War, an American Patriot Missile battery in Dharan, Saudi Arabia, failed to intercept an incoming Iraqi Scud missile. The Scud struck an American Army barracks and killed 28 soldiers.

Patriot

"The range gate's prediction of where the Scud will next appear is a function of the Scud's known velocity and the time of the last radar detection. Velocity is a real number that can be expressed as a whole number and a decimal (e.g., 3750.2563...miles per hour). Time is kept continuously by the system's internal clock in tenths of seconds but is expressed as an integer or whole number (e.g., 32, 33, 34...). The longer the system has been running, the larger the number representing time. To predict where the Scud will next appear, both time and velocity must be expressed as real numbers. Because of the way the Patriot computer performs its calculations and the fact that its registers are only 24 bits long, the conversion of time from an integer to a real number cannot be any more precise than 24 bits. This conversion results in a loss of precision causing a less accurate time calculation. The effect of this inaccuracy on the range gate's calculation is ***directly proportional to the target's velocity and the length of the system has been running****. Consequently, performing the conversion after the Patriot has been running continuously for extended periods causes the range gate to shift away from the center of the target, making it less likely that the target, in this case a Scud, will be successfully intercepted."*

Government Accounting Office Report http://www.fas.org/spp/starwars/gao/im92026.htm

Patriot

- This bug is typical of a requirements deficiency caused by reuse
- Patriot was originally an anti-aircraft (防空) system designed to remain “up” for short periods of time and to track slow (~mach 1-2) targets
- It was moved into a missile defence (反飛彈) role where it now had to be on station for many days and to track much faster targets