Is Software Really the Weak Link In Dependable Computing?

Gérard Le Lann

INRIA

Gerard.Le_Lann@inria.fr

Simplified Life Cycle

- Application requirements capture ⇒ <X>
- System design ⇒ Design solution [Δ] + Proofs (validation)
- System dimensioning (Proofs used):

  dimensioning of <X> ⇒ dimensioning of [Δ]

  ---------- System Eng. (SE) ↗↗↗ ----------

  ---------- S/W + H/W Eng. ↘↘↘ ----------

- System implementation ⇒ S = H/W + S/W implementation of [Δ]

  (Design + Proofs (verification))

- Fielding + use of S

Proof Obligations with [Δ]

[Δ] solves <X> = proofs that P hold, under A and some design assumptions, for specific feasibility conditions (FCs) – to be established (e.g., nb. of processors > 2t, external "load" < L) **- with some coverage meeting a specified lower bound.**

Example: Timeliness ("real-time") FCs. Overlooked obligation, too often, which explains (quasi) failures (see Mars Path Finder).

Timeliness FCs?

Δ must include some scheduling algorithm(s) Sched
(e.g. HPF, EDF, SSF, D-Over, …).

Proofs of Timeliness? $\Rightarrow$ Schedulability analyses:

(1) Identification of (worst-case, average case, …) scenarios that can be deployed by adversary A in the presence of Sched,

(2) Establishment of a set of constraints (FCs) $\Rightarrow$ analytical expressions of Timeliness achieved by Sched in the presence of A.

Real problem RX ⇒ requirements capture ⇒ **<X> = {<A>; <P>}.**

A = assumptions = an "adversary" = expected run-time conditions (environment, technology, ..), which cannot be "simplified".

Typically: **A** = {events arrival, process, failure, …} models.

A should be fully and completely specified. Overlooked obligation, too often, which explains failures.

Notably: Variables shared by (future) system S and A, or their values. See A5/501, Mars Climate Orbiter, Mars Polar Lander, Titan IV/Milstar.

Why (too many) operational failures?

What is blamed, usually? The S/W!

(Faults left, not "enough" testing, not "enough" formalism, etc., regarding S/W design and/or implementation)

Reality?

A5/501

(faulty requirements capture)

European satellite launcher Ariane 5, maiden flight 501 (June 1996).

Blew up after 37 seconds. Loss ≈ 0.5 Billion US$.

Official diagnosis (Inquiry Board Report): "It's the software". (Overflow of a variable (HB) while running a program (RP) that serves to realign the inertial platform).

Real cause (source node of the causal graph): Faulty (incomplete) requirements capture – see the Safety-Critical Forum.

A5's horizontal velocity up to 5 times A4's horizontal velocity. Data known to A5 designers. Never "captured" by the designers of the on-board system (a SE fault).

HB's numerical value was correctly computed by RP, but was larger than can be accommodated with the dimensioning of A4 on-board system – reused unchanged for A5! (a SE fault).

Moreover … program RP serves no purpose with A5.

Had RP been implemented in tupperware, and the same "capture" fault made, flight 501 would have failed. S/W cannot be the cause of the failure.

US Mars Climate Orbiter, Sept. 1999 (1), Mars Polar Lander (2),
Titan IV B-32/Centaur TC-14/Milstar-3, April 1999)* (3), …
failed for similar reasons.

(1) Faulty (sub)system dimensioning (portion of trajectory model):
Metric units ≠ English units.

(2) Faulty requirements capture:
False momentary signals from touchdown sensors. Known phenomenon.
Was not captured (System engineers produced an incomplete <A>).

(3) Faulty (sub)system dimensioning (incorrect roll rate filter constant):
Should have been set to -1.992476 (exponent = 1),
instead of 0,1992476 (exponent = 0).

* Cost ≈ 1.23 Billion US$

Danone vs. Cegelec case

(faulty system design)

Dairy products factory (late 80's). Brand new computer-based system S never accepted by Danone (nor paid) ⇒ a 2.5 year long legal case.

Official diagnosis: "It's the software".

New application S/W "too complex", said Cegelec, who needed more time, more money, to "clean it up" ☺

Real causes (we were asked to look at it): S did not include essential algorithms/protocols – e.g., no mutual exclusion! (**a SE fault**).

Mars Path Finder

(faulty system dimensioning)

JPL has rescued the mission (1996), via remote testing and correction.

Official diagnosis: "It's the software".

Some task was missing its 125 ms deadline repeatedly.

Cause: The priority-inheritance (PI) option with VxWorks was set to "off" (had to be set "on").

Real cause: No Timeliness FCs established (**a SE fault**).

With Timeliness FCs at hand – a set of computable constraints – it would have been possible to check, before launch, that:

- With PI "off": the 125 ms deadline is missed,
- With PI "on": the 125 ms deadline is met.

What has S/W to do with this? Nothing! Had the PI option been implemented as a mechanical switch, same outcome…

Blaming the S/W is like blaming the engine of your car, because your car is too slow. Real cause: You drive in first gear all the time (wrong option, the engine is not "faulty").

CONCLUSIONS (1)

► **Infatuation with S/W considered harmful.**

► **We should pay more attention to SE issues.**

► **More proofs, more "science" (architecture, algorithms), should underlie SE practice ⇒ Proof-Based SE.**

Are we (scientists) going to meet the challenge of helping our friends in industry to change their views on current practice? (We are being paid for doing knowledge transfer).

CONCLUSIONS (2)

If you need an accident to know there is a problem, then you are part of the problem (Joe Barton).

Proof-Based SE is taking off.

Since 1996, TRDF, a proof-based SE method, has been tested or used in various areas – e.g., avionics, nuclear power plants, space, air traffic control – by, e.g., DGA/DSP (French Darpa), CNES (French Space Agency), ESA/ESTEC (European Space Agency), Dassault Aviation, EADS LV, Astrium, Axlog, Thales Airsys.

CONCLUSIONS (3)

If you need an accident to know there is a need for continued research, then you are doing the human gender a big disservice.

Are we (scientists) going to meet the challenge of focusing our research work on problems and assumptions in dependable computing that meet reality better (than the case up to now)?