

Building systems from unreliable components

Breakout Group of Dagstuhl Seminar 04511 12/2004

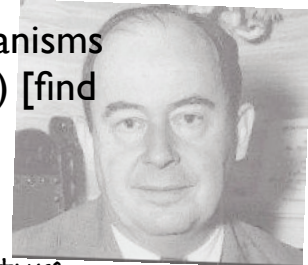
Heinz W. Schmidt,
Monash University

Joint work with:

Judith Stafford, Nicole Levy, Ivica Crnković, Rob van
Ommering, Shriram Krishnamurti

www.csse.monash.edu.au/~hws

Probabilistic logic and the synthesis of reliable organisms from unreliable components. Von Neumann (1952) [find also in 'Collected Works' 1956]



- one of the first papers on component-based architecture



Concerns

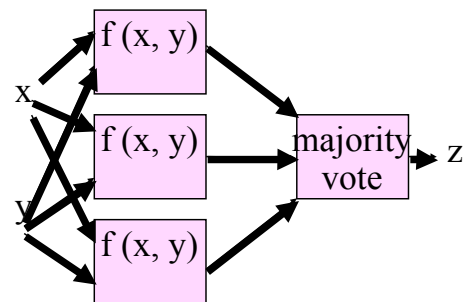
- ENIAC ~17,500 vacuum tubes
MTTF days to months
- Static defects: bad products or material fatigue
- Dynamic defects: transient failures or intermittent failures

Von Neumann's Solution

- Mathematical logical theory of reliability
 - Reliability ~ probability of non-failure
 - Failure ~ execution not meeting expected functionality
 - Fault ~ defect causing failure
- Engineering approach
 - Fewer parts
 - Better parts
 - Redundancy, s.a.:

Parallel restitution:
1 wire -> n bundle

Triple modular redundancy



Unreliable Components - Heinz Schmidt - Dagstuhl Seminar 04511 - 12/2004

Numerous examples for low-reliability parts in reliable systems at any scale

- Networks (OSI/ISO layers):
 - no guarantee of delivery -> delivery but no message order guarantee -> highly reliable service guarantee
- Human brain (as already observed by von Neumann)
 - reallocation of brain functions after seizures, bypassing 'failing components'
 - parallel function on top of statistically 'unreliable' neuronal signal processing each with different (low) reliability in function and timing
- Reliable consumer electronics (e.g. Philips):
 - large-scale components with transient/intermittent MTTF in days/weeks: fast rebooting OK now, but with increasing ratio SW/HW reboot times may be prohibitive
- Volvo construction vehicles
 - ditto, fail-reboot some devices in operation within ms to s range

Unreliable Components - Heinz Schmidt - Dagstuhl Seminar 04511 - 12/2004

What is going on?

End-To-End Arguments In System Design (1984) (381 citations [Citeseer])
Jerome H. Saltzer, David P. Reed, David D. Clark, ACM Transactions on Computer Systems

“The principle (end-to-end argument) suggests that functions placed at low levels of a system may be redundant or of little value when compared with the cost of providing them at that low level. Examples ... bit error recovery... duplicate message suppression... delivery acknowledgement. Low level mechanisms to support these functions are justified only as performance enhancements.”

1. Small low-level extra can go a long way in high-level guarantees
2. Lower levels need not provide high reliability; in fact, big footprint low-level reliability (~100%) is prohibitive for high-level guarantees
3. Low level extras need to be fast and small (so that reliability gains, at higher level, from retries and redundancy in low-level function use are not waived by performance loss resulting from such redundancy).

Shriram: isn't it weird and notable that performance is essential for functionality!

Unreliable Components - Heinz Schmidt - Dagstuhl Seminar 04511 - 12/2004

Suspicious reboots:

Is it acceptable to bypass faults when we should really remove them?

■ Recovery-oriented computing

[Patterson, ROC: A research agenda for a new century OS]

“Traditional Fault-Tolerant Computing concentrates on tolerating hardware and operating system faults-- Recovery Oriented Computing (ROC) aims at improving Mean Time To Recover to...improve the availability of whole system...”

■ Observation:

- long runs of complex systems increase the likelihood of reaching unsafe states (buffer overruns, memory access faults) risking failure and security breaches

■ Remedy:

- proactive restarts at safe points (after save) to get from 'risky' to 'safe' states
- fast-restart support on chip and in OS

Unreliable Components - Heinz Schmidt - Dagstuhl Seminar 04511 - 12/2004



Recovery-Aware Components

- *RCC CB Services*
 - *underlying framework recovery-oriented w/ restart signals, fast restart, fault-bypass and light-weight logs*
 - *Recovery-aware components API: save-restart-selftest handlers*
- *Types of faults and benefit:*
 - *Context: components participate improving reliability and availability*
 - *Configuration: service ranking and self-test for fast bypass and graceful degradation of service*
 - *Component: recovery logs provided to developers for improved fault detection*
- *AO recovery modeling and prediction*
 - *Weaving fail-over, bypass, restart with component protocols for modeling component-framework fault-tolerance interaction and predicting availability*
 - *Starting points: probabilistic state machines and FT controller synthesis coordinating low-reliability services [PRISM, PCTL, SAVCBS'03 paper]*

Unreliable Components - Heinz Schmidt - Dagstuhl Seminar 04511 - 12/2004

Faulty systems built from reliable components

Emerging global failures: 'Blame the integrator'?
Global fault analysis/test?



- *Faults resulting from composition or global interaction, for example:*
 - *cyclic dependencies breaking coherence;*
 - *components used in contexts they were not made for;*
 - *local coherence, compatibility and greedy recovery often insufficient*
- *1990 ATT network crash where auto-rebooting of faulty routers (SW) under load led to increasing occurrences of same failure in more and more routers across network ultimately taking network down*

Unreliable Components - Heinz Schmidt - Dagstuhl Seminar 04511 - 12/2004