

55th IFIP WG 10.4 Meeting

Cortina d'Ampezzo, Italy – January 27-31, 2009

Workshop on Evaluation of Dependability and Resiliency

From Fault Injection-based Assessment to Dependability Benchmarking: A 4-Way Street or a Long and Winding Road?

Jean Arlat

[jean.arlat@laas.fr]



Université
de Toulouse



Outline

- **Experimental Dependability Assessment**
- **Fault Injection Experiments**
 - ◆ The {FARM} attributes
 - ◆ The techniques
 - ◆ Coverage estimation
 - ◆ Some lessons learned
- **Dependability Benchmarking**
 - ◆ Rationale and objectives
 - ◆ Desired properties
- **Looking Ahead and Challenges**
 - ◆ Faultload
 - ◆ Injection techniques
 - ◆ Objectives and measures

About Coverage

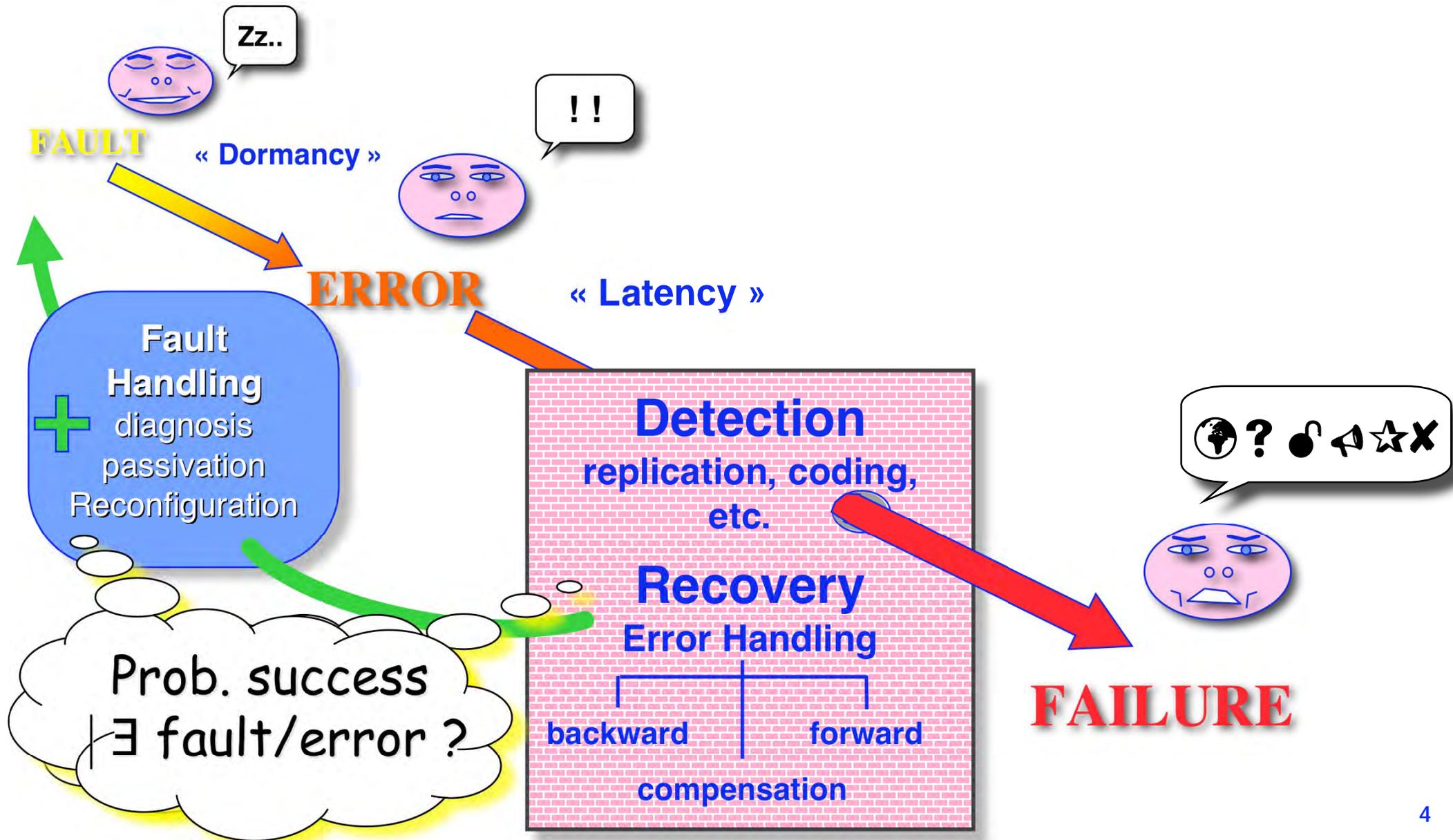
W. G. Bouricius, W. C. Carter and P. R. Schneider
Reliability Modeling Techniques for Self Repairing Computer Systems
Proc. 24th. National Conference, pp.295-309, 1969.

... Define the coverage c to be the conditional probability that, given the existence of a failure in the operational system, the system is able to recover, and continue information processing with no permanent loss of essential information, i.e.,

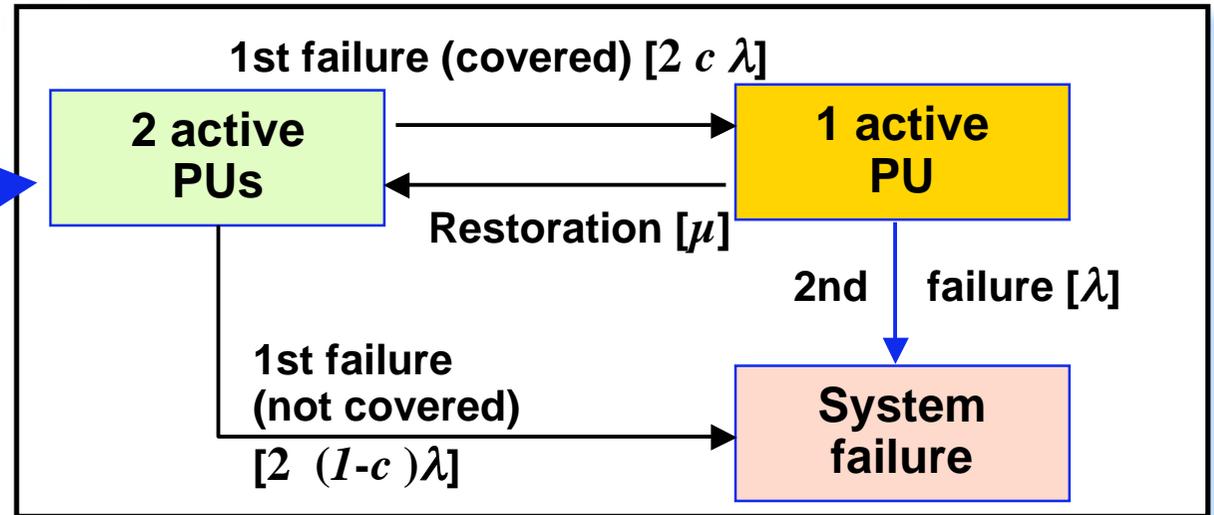
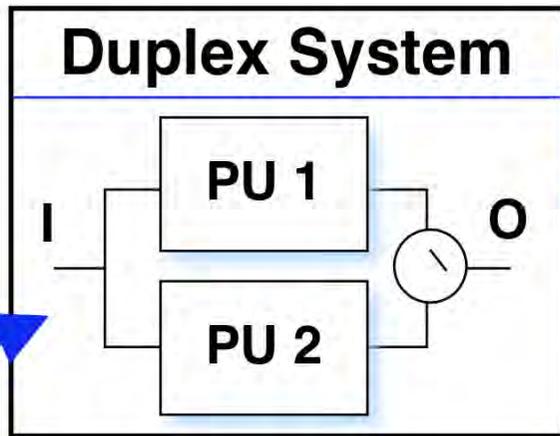
$$c = \text{Pr} [\text{system recovers} \mid \text{system fails}].$$

Exactly what constitutes recovery is a matter for the individual system designer to settle; at this point it is just a system parameter. In some situations recovery may only mean detection, ...

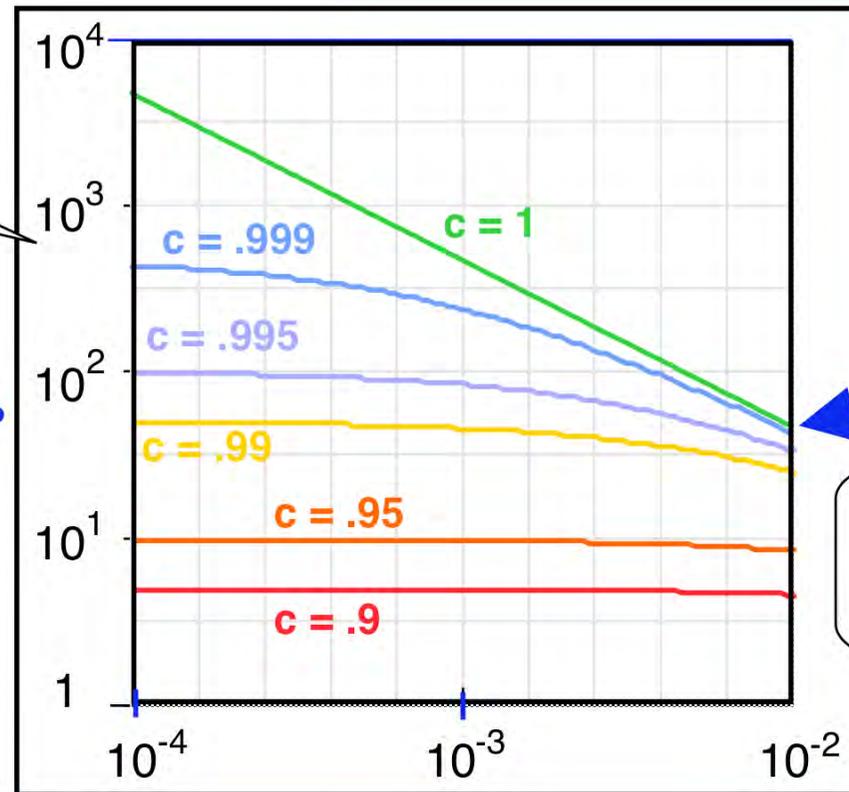
Fault Tolerance ... and Coverage



Impact of FT Coverage on Dependability



$$\frac{\text{MTTF}_{\text{DS}}}{\text{MTTF}_{\text{PU}}}$$



$$\frac{\text{MTTR}_{\text{PU}}}{\text{MTTF}_{\text{PU}}} \left(\frac{\lambda}{\mu} \right)$$

A Rationale for Experimental Assessment

Dependability



Fault Tolerance (FT)

- FT mechanisms = human artefacts (not perfect)



- Impact on dependability measures

- Calibration of models



- Estimation of FT coverage

- Formal approaches limits



- Experimental approaches

- Fault/threat = rare event



- Controlled experiments



Fault Injection



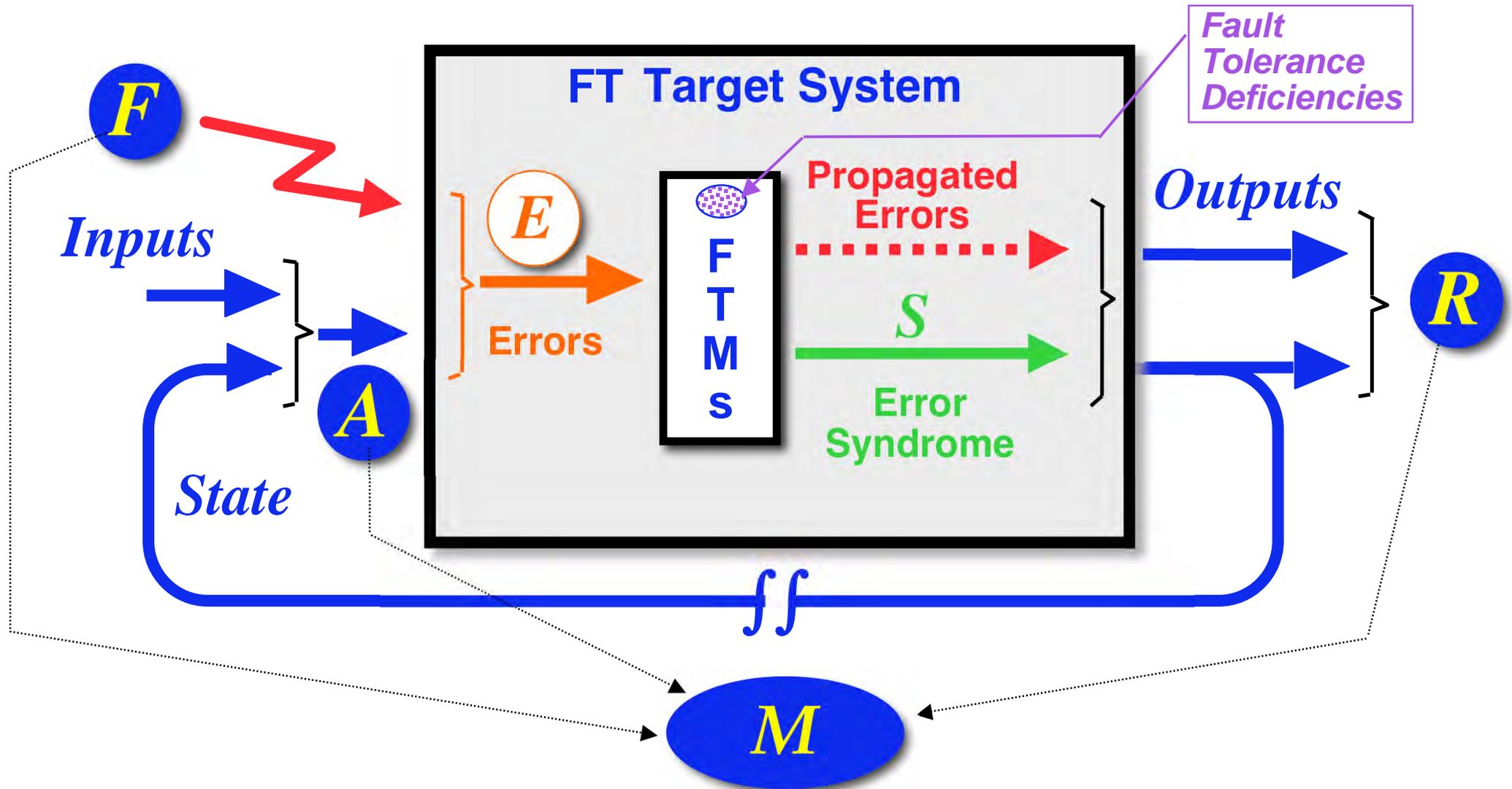
"Assessment of fault tolerance
wrt specific inputs it is designed to deal with: *the faults*"

Fault Injection-based Assessment

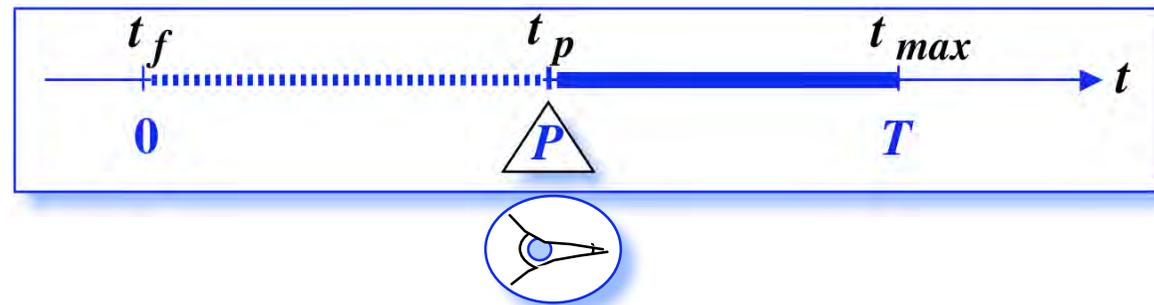


- Testing and evaluation of a fault-tolerant system and of its FT algorithms & mechanisms
 - Partial dependability assessment: controlled application of fault/error conditions

The Fault Injection Attributes

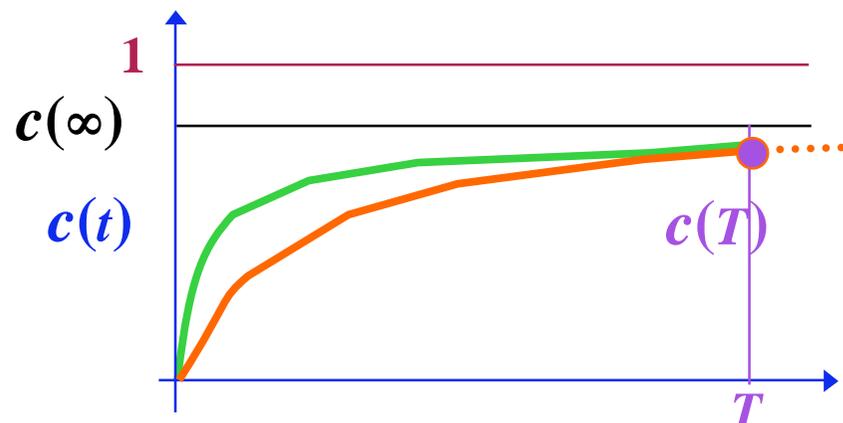


A Typical Fault Injection Experiment



Experiment \approx
Bernoulli trial

- Observation of FT TS reaction/behavior $r \in R$ when subjected to fault $f \in F$ in presence of activity $a \in A$
- Series of experiments \rightarrow descriptive statistics & measures \rightarrow Inferential stats on coverage: $c(t) / \{F, A\}$?



■ Examples of properties/predicates

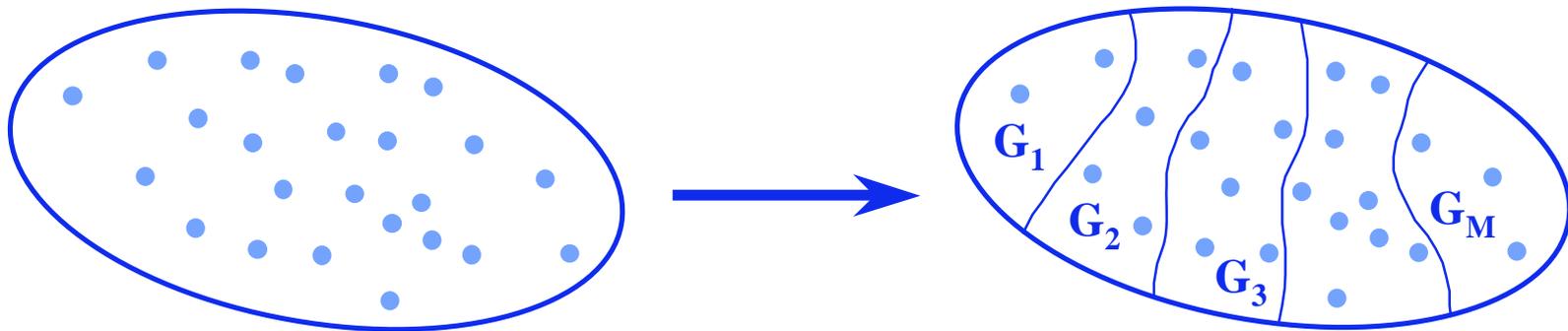
- ◆ D (detection) \rightarrow conservative estimate?
- ◆ T (recovery) \rightarrow optimistic estimate?

Simple Sampling → Stratified Sampling

- The fault-activity set is partitioned into classes

Fault-activity set

$$G = F \times A$$



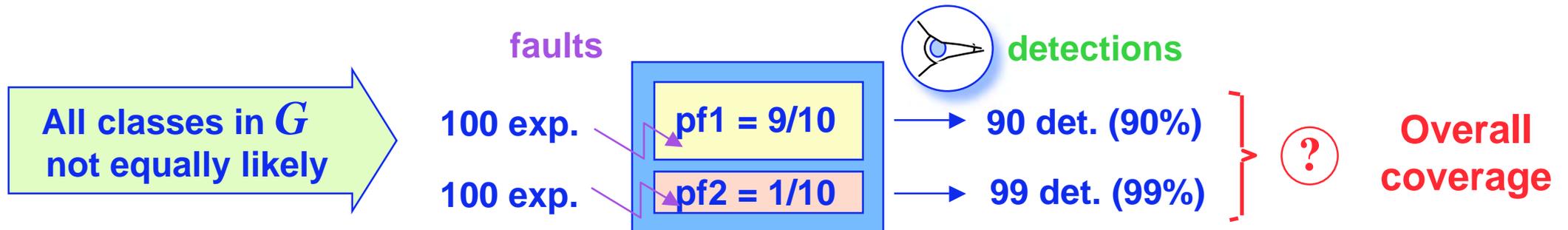
- Several opportunities
 - ◆ Transient, intermittent, permanent faults
 - ◆ Activity/Workload profiles
 - ◆ TS components
 - ◆ ...

D. Powell, E. Martins, J. Arlat, Y. Crouzet

Estimators for Fault Tolerance Coverage Evaluation

IEEE TC (Special Issue on Fault-Tolerant Computing), 44, (2), pp.261 - 274, Feb. 1995

Estimation of Asymptotic Coverage



Choice of an estimator:

- Stratified sampling, representative sample per strata and weighted estimator
 -> unbiased estimation of coverage for classical systems

$$\hat{C}_2(G) = \sum_{i=1}^M p(G_i|G) \cdot \hat{C}_1(G_i) = \sum_{i=1}^M p(G_i|G) \cdot \frac{N_i}{n_i} = \frac{1}{n} \sum_{i=1}^M \frac{p(G_i|G)}{t(G_i|G)} \cdot \sum_{j=1}^{n_i} y(g_j)$$

"real" distr.

sampling dist.

OK, but, what about highly dependable systems? (high coverage requirement)

—> [Frequentist vs. Bayesian] stats based on "Confidence Region" theory

Some Milestones: The Early Years...

- Late 60s & 70s: FI exp. on major FT computer systems
 - ◆ *STAR* (JPL & UCLA), *FTCS* (Raytheon),...
- Late 70s: Code mutation for SW testing
- Early 80s: Pin-level FI technique
 - ◆ MSI FI chips (Spaceborne Inc)
 - ◆ Insertion → Forcing : *MESSALINE* (LAAS)
- Late 80s:
 - ◆ Heavy-ion radiation (Chalmers U)
 - ◆ The *FARM* FI attributes (LAAS)
 - ◆ Compile-time SWIFI : *FIAT* (CMU)
 - ◆ Failure Acceleration concept (IBM)
 - ◆ Hierarchical Simulation (UIUC)
- Early 90s: FI in VHDL models
 - ◆ Petri Net-based simulation (U. Virginia)
 - ◆ *Saboteur*-based FI: *MEFISTO* (Chalmers U+LAAS)
- Mid 90s: Run-time SWIFI
 - ◆ *FERRARI* (U Texas), *Xception* (U Coimbra), ...



Hardware + SW

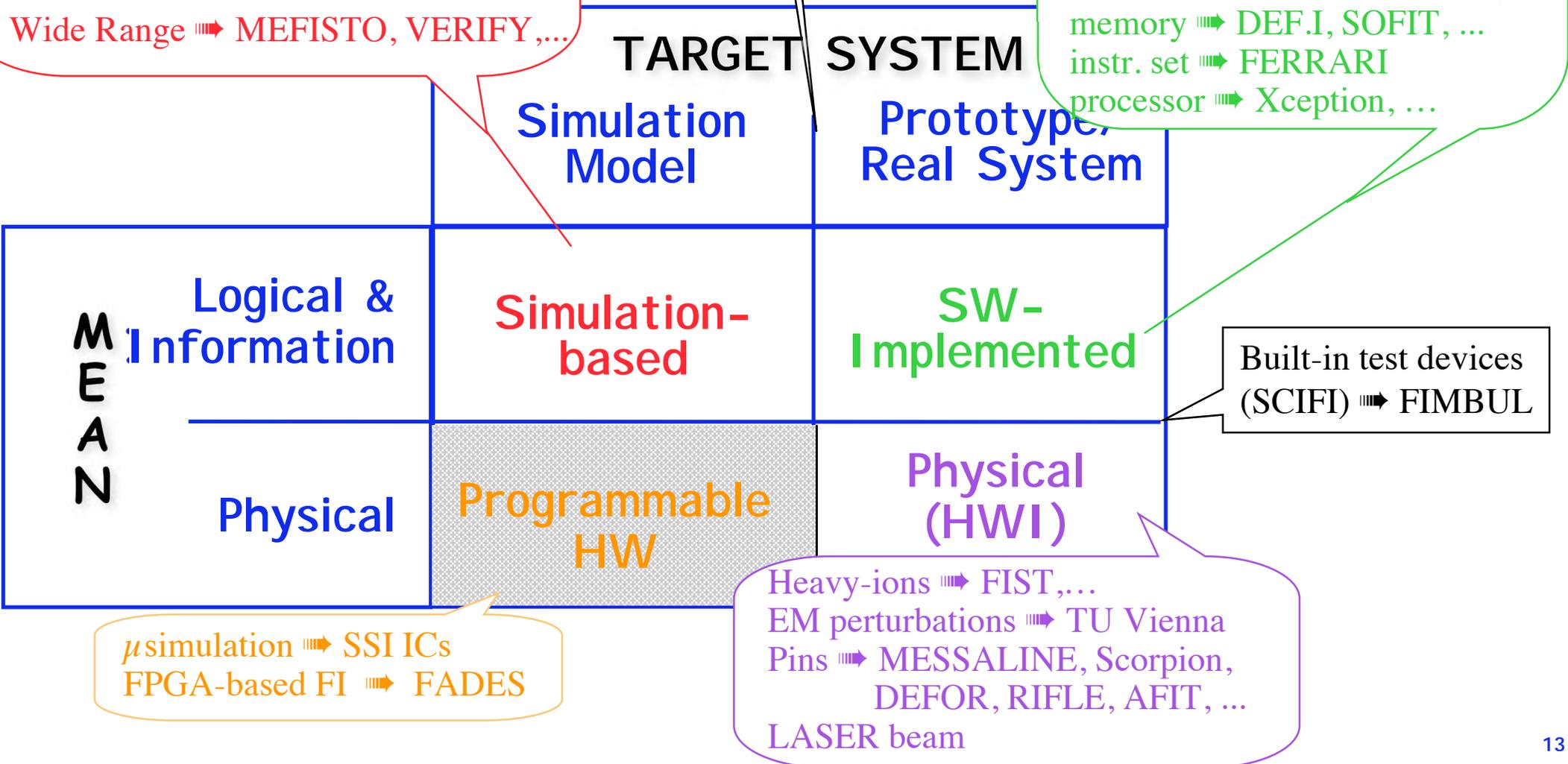
The Fault Injection Techniques

system \Rightarrow DEPEND, REACT, ...
 RT Level \Rightarrow ASPHALT, ...
 Logical Gate \Rightarrow Zycad, Technost, ...
 Switch \Rightarrow FOCUS, ...

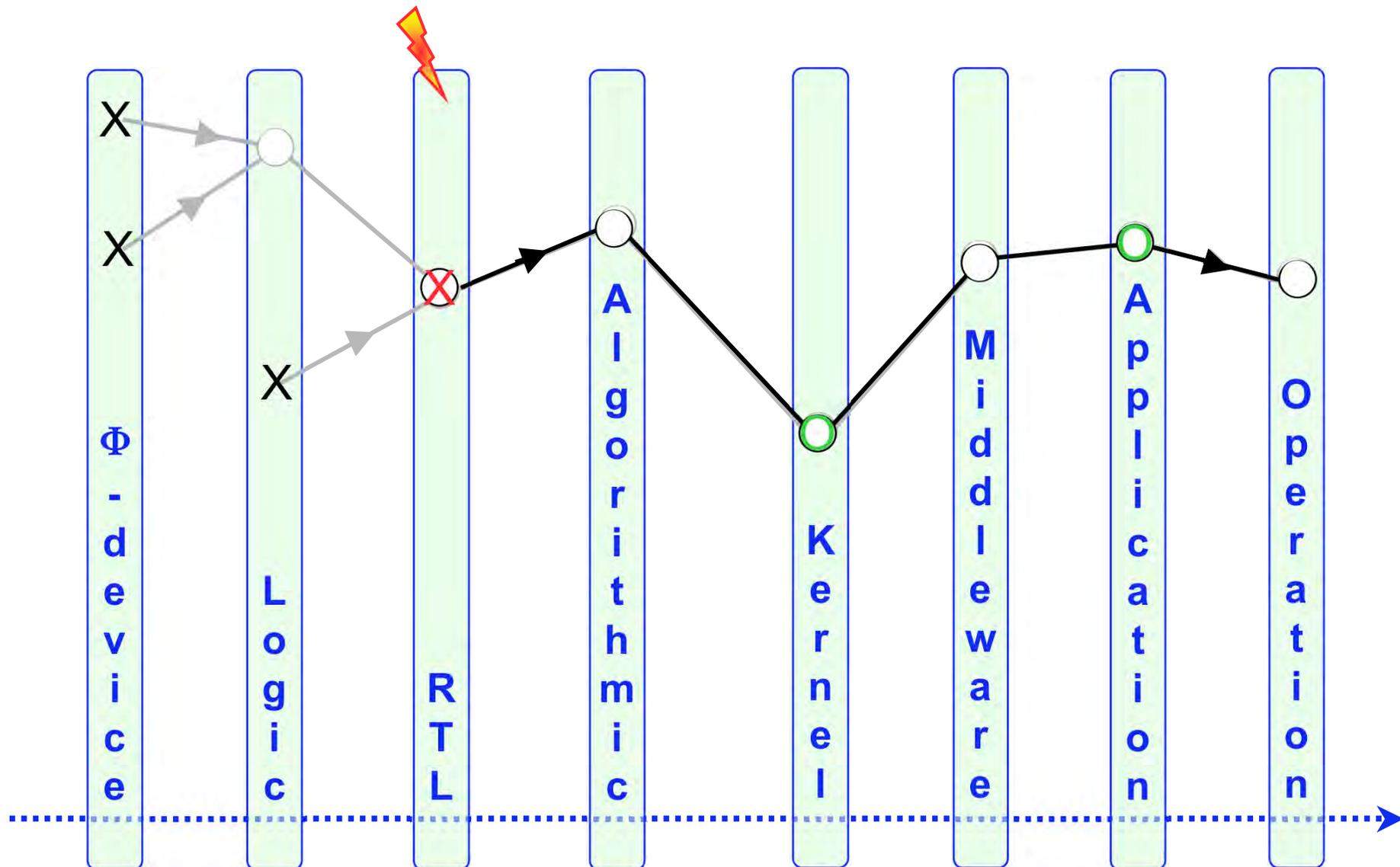
Wide Range \Rightarrow MEFISTO, VERIFY, ...

Compile-time
 software mutation
 \Rightarrow SESAME, G-SWFIT

communication \Rightarrow ORCHESTRA
 node CoFFEE
 debugger \Rightarrow FIESTA
 task \Rightarrow FIAT
 executive \Rightarrow Ballista, (DE)FINE,
 MAFALDA-RT,
 memory \Rightarrow DEF.I, SOFIT, ...
 instr. set \Rightarrow FERRARI
 processor \Rightarrow Xception, ...

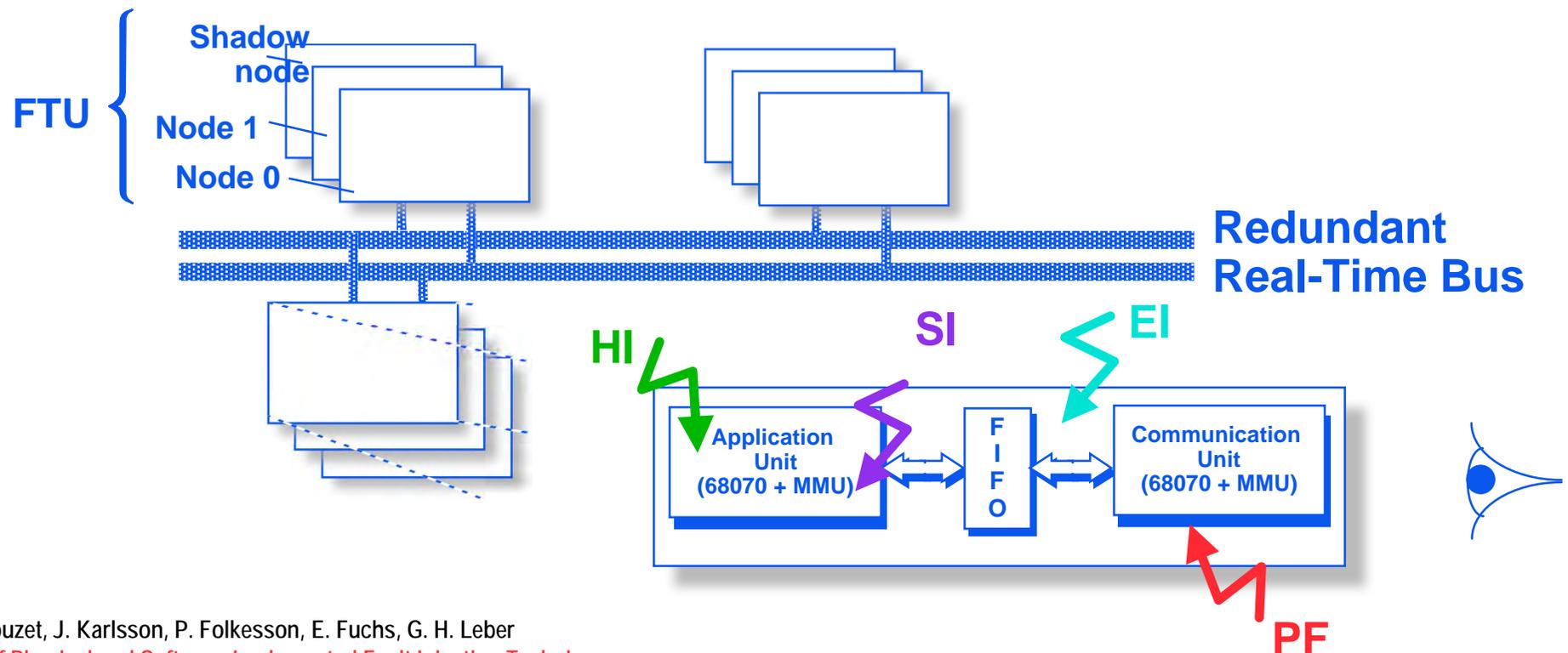


Target System Levels & Fault Pathology



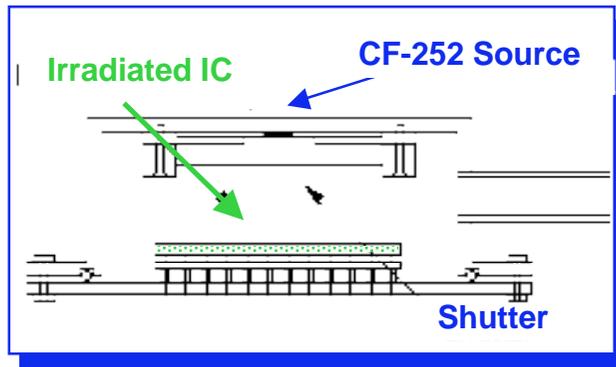
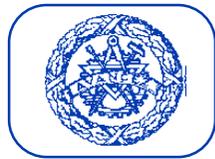
FI Experiments on MARS: Dual Objectives

- Extensive Assessment the "Building Block" of the **MAintainable Real-time System (MARS)** FT Architecture: *the Fail-Silent Node*
- Compare the 4 Fault Injection Techniques Considered (**Heavy-Ion radiations, Pin-Forcing, EMI** and **CT-SWIFI**)

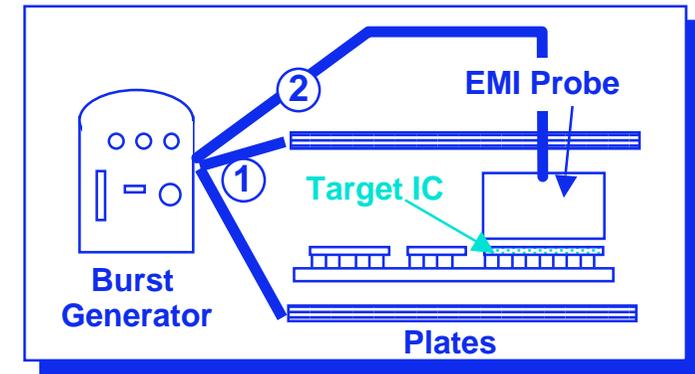


The Fault Injection Techniques

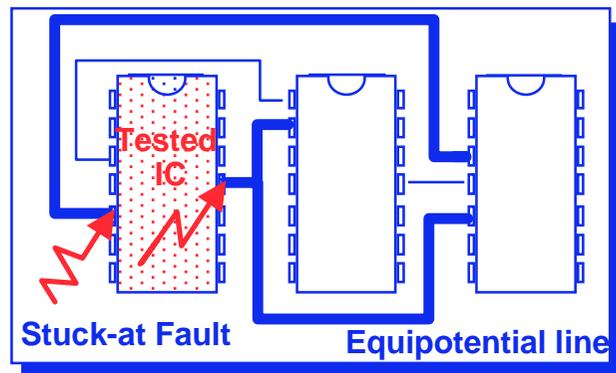
- Heavy-Ion Radiation (HIR)
 - + Reachability (Internal IC faults)



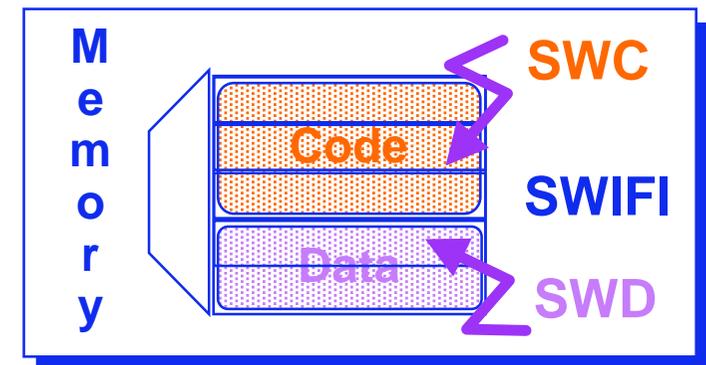
- Electro-Magnetic Interference (EMI)
 - + Flexibility (adaption to several systems)



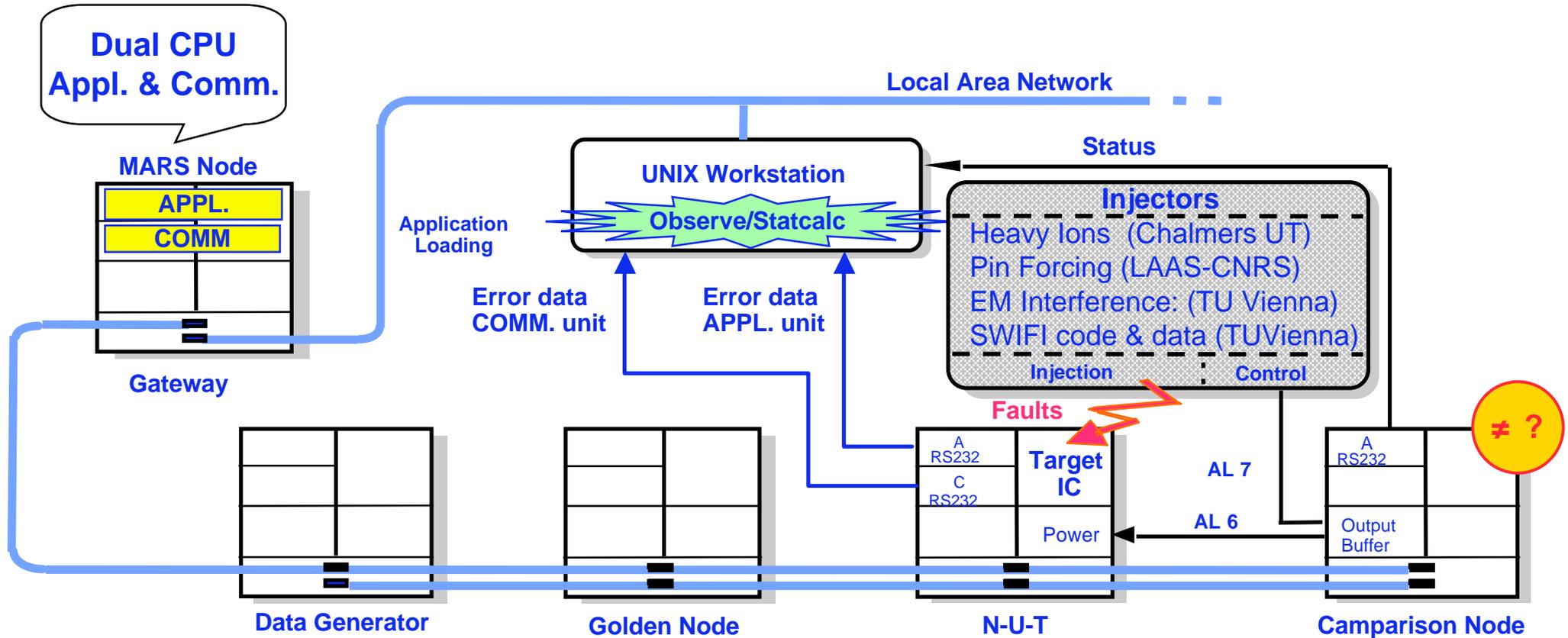
- Pin-level Injection by Forcing (PIF)
 - + Controllability (distribution among ICs, timing)



- Software-Implemented Fault Injection (Compile Time)
 - + Ease of application



The Testbed



The Error Detection Mechanisms (EDMs)

■ Level 1 — Hardware

- ◆ CPU: Bus Error, Address Error, Illegal Opcode, Privilege Violation, Zero Divide, etc.
- ◆ NMI: W/D Timer, Power, Parity, FIFO Mngmt, Memory Access, NMI from other Unit, etc.

■ Level 2 — Software

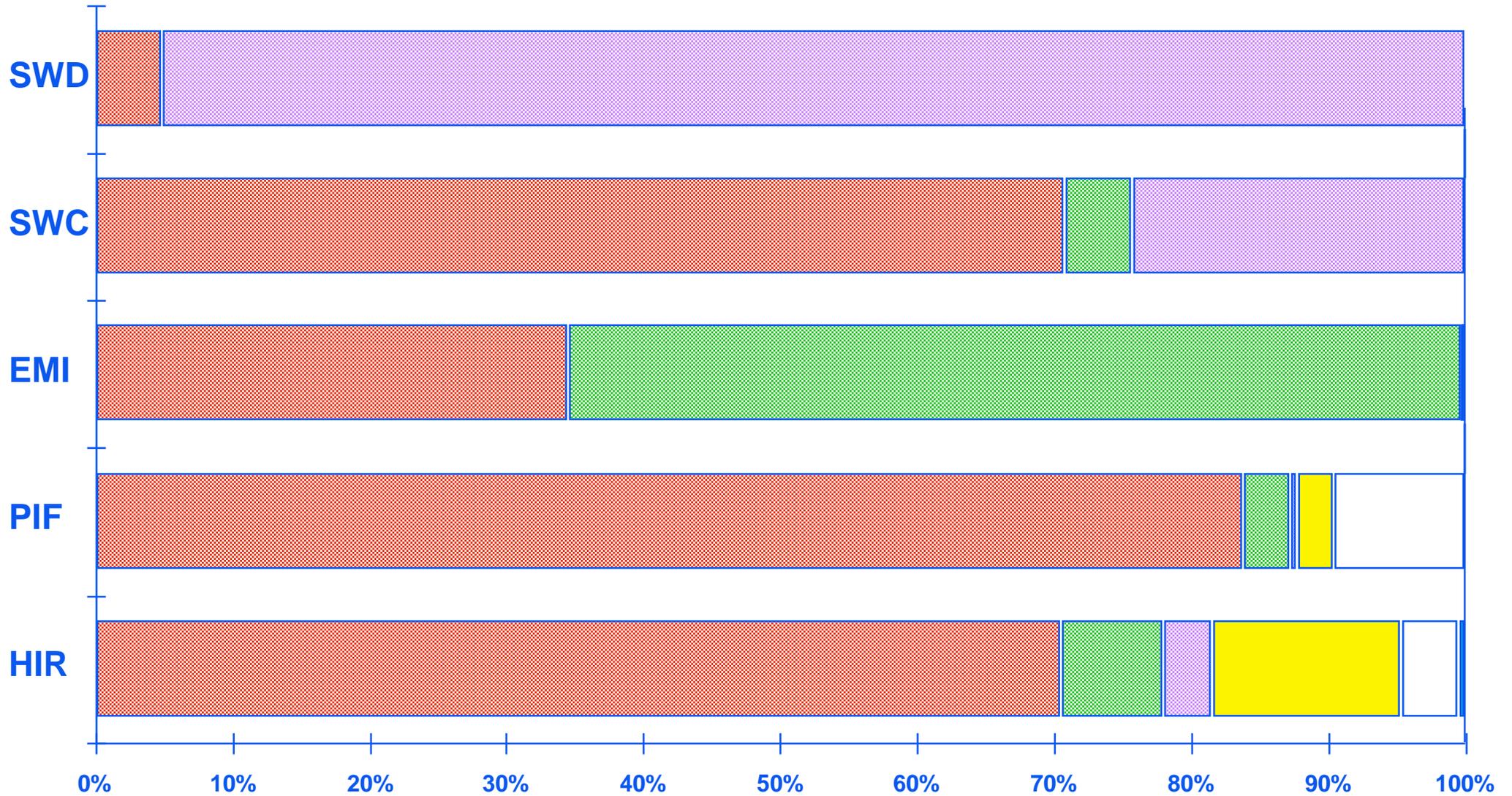
- ◆ Operating System (OS): Processing time overflow, various assertions in the OS, etc.
- ◆ Compiler Generated Run-Time Assertions (CGRTA): Value range overflow, etc.

■ Level 3 — Application

- ◆ Message Checksum
- ◆ Double Execution (Checksum Comparison)

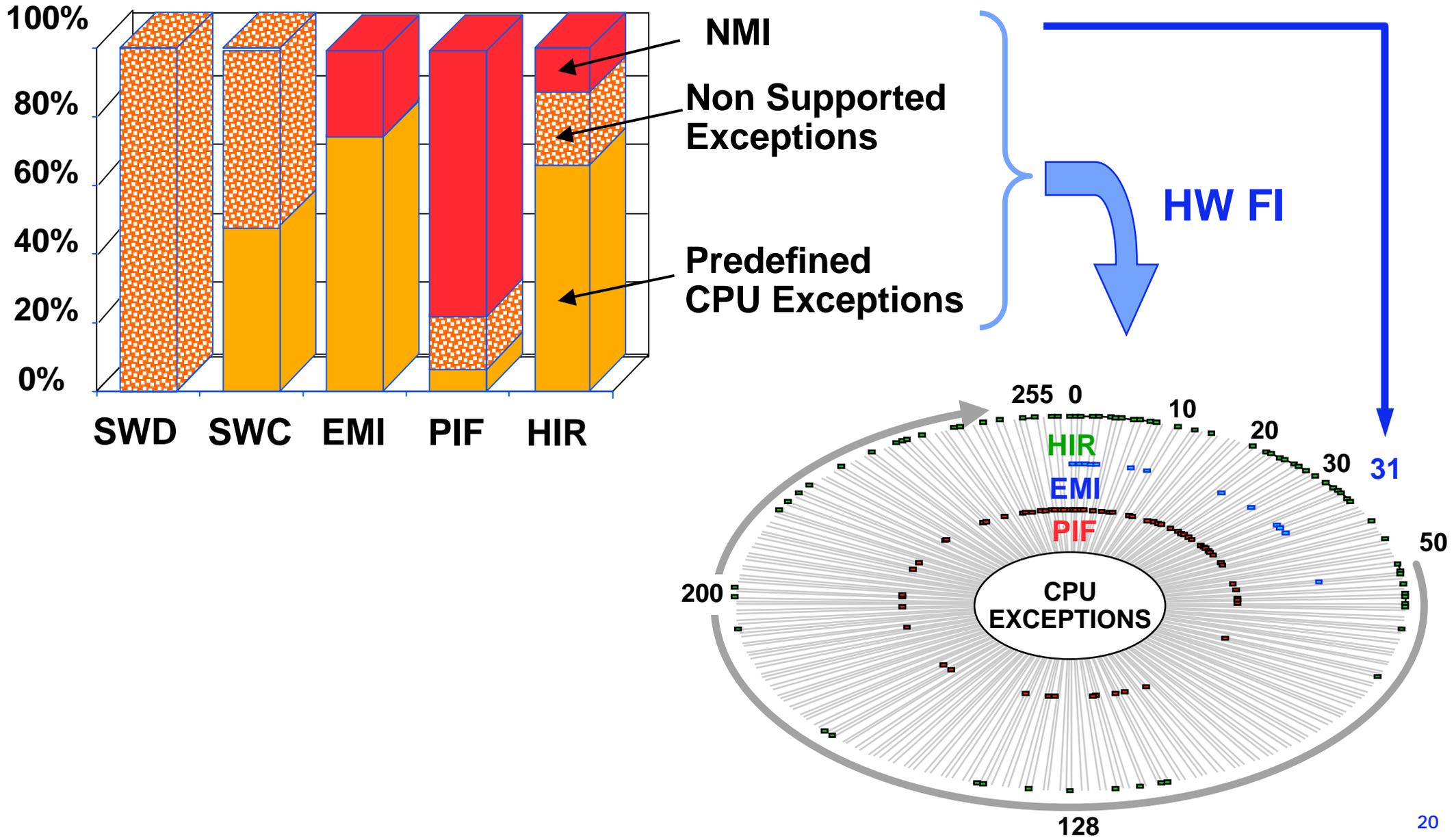
Error Distributions

[All Error Detection Mechanisms Enabled]



Detailed Contribution of HW EDMS

[All EDMs Enabled]



Some Milestones: More Recent Years...

■ Late 90s: *En-route* to Dependability/Robustness Benchmarking

- ◆ API-based FI: the *CRASH* scale and *Ballista* tool (CMU)
- ◆ SW μ ernels: *MAFALDA* (LAAS)
- ◆ BIST-based FI *FIMBUL* (Chalmers)
- ◆ IFIP WG. 10.4 SIG DeB

■ Early 00s: IST Project DBench

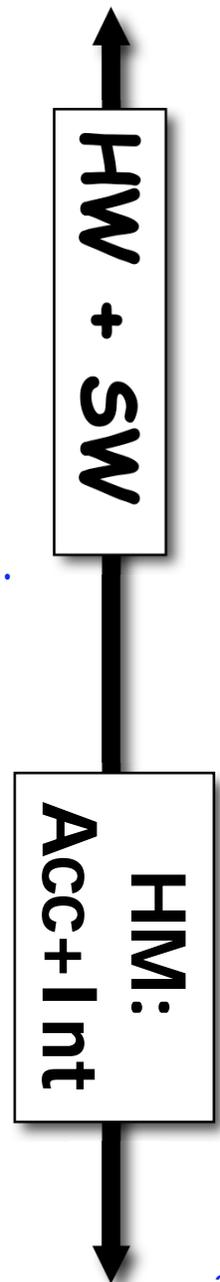
- ◆ SW Executives: OS (*DBenchOS-API*, *Rocade-DPI*), Corba (*CoFFEE*), ...
- ◆ Databases & Web services: *OLTP-Bench*, *G-SWFIT* (U Coimbra)
- ◆ Embedded systems: (PU Valencia, Erlangen U., *DeBERT* Critical SW)

■ Mid 00s:

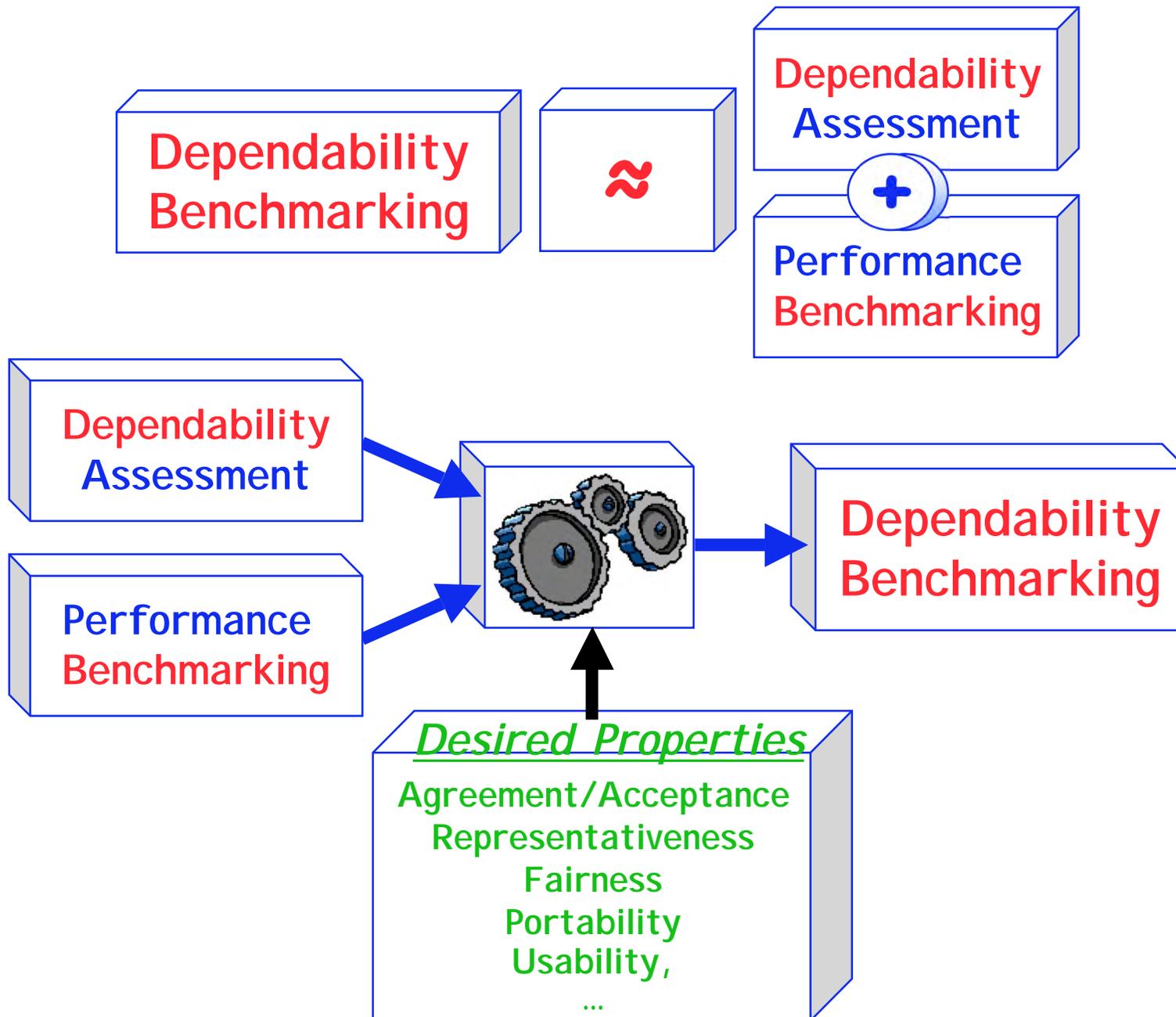
- ◆ Threats targeting **vulnerabilities** <-> **security** (UIUC, U Coimbra, U Leeds, TIMA, U Marseille,...)
- ◆ **FPGA-based FI** : *FADES* (PU Valencia,...)
- ◆ Human/Operator errors: CMU, U. Coimbra, *ConfErr* (EPFL), ...

■ Late 00s:

- ◆ Assessment of Intrusion Detection Systems (IBM, LAAS,...)
- ◆ Book on *Dependability Benchmarking* (WG 10.4 SIG DeB + DBench)



Dependability Benchmarking



FI Campaign vs. Dependability Benchmark

FTS Assessment

- 1 Target System
- In-Deep Knowledge OK
- Fault and Activity sets
- Sophisticated (intimate) faults
- Measures = Conditional dependability assessment
- FTMs testing
- One-of-a-kind process: "heavy weight" still OK
- Developer's view

Dependability Benchmarking

- > 1 Target Systems [Components]
- Limited Knowledge only
- Fault- and Work-load
- Reference (interface) faults only
- Measures = Dependability assess. —> Fault occurrence process
- Global behavior
- Recurring process: "user friendly" required
- End User/Integrator's view

Common Properties

Non Intrusiveness: No temporal behavior affectation nor target system alteration

Representativeness: Fault and Activity/Work set/loads

Repeatability: Obtention of statistically equivalent results

“Benchmark-Specific” Properties

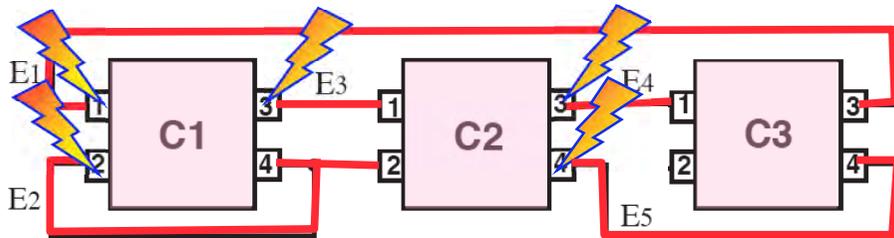
- **Portability:** Applicability to various Target Systems
 - **Reproducibility:** Ability for another party to run the benchmark and obtain statistically equivalent results
 - **Usability:** Ease of installation, running and interpretation
 - **Fairness:** Comparisons made should rely on equitable assessments
 - **Scalability:** Applicability to evolving Target Systems e.g., configuration changes, etc.
- > **Agreement** on procedures, and disclosure & publication policies

Some Advances and Challenges

- About the F set: focus (reduce) the F set
- Improve the effectiveness (testing capabilities) of the FI experiments → pre-analysis (F & A sets)
- Fault injection techniques: Hardware-level fault injection?
- Dependability benchmarking: agreement about FI interfaces and R & M sets
- Derivation of dependability measures
- Security: vulnerabilities and attacks
- Openness: highly interactive systems (incl. embedded systems)
- Evolvability: high change rate of TS → on-line assessment
- Usability, Scalability, ...

Managing the size of the F set

- **HWIFI**: Analysis of the connection list (MESSALINE)



- **SWIFI**: Analysis of the SW code (GOOFI)

$R1 + 16 \rightarrow R2$

$R1 + 12 \rightarrow R1$

$17 \rightarrow R3$

→ $R2 + R3 \rightarrow R4$

→ $R1 + R2 \rightarrow R3$

$R3 + R4 \rightarrow R2$

} Valid points
for FI
in $R2$

Other applications of "fault collapsing"

- Assembly code [Benso *et al* 98]
- VHDL models [Berrojo *et al* 02]

Path- & stress-based FI [Tsai *et al* 99]

- ◆ Increase of 1 order of magnitude in the "effectiveness" of faults
- ◆ Reduction of the F set:
2 orders (CPU reg.); 4-5 (data mem.), still with similar estimation of coverage

R. Barbosa, J. Vinter, P. Folkesson, J. Karlsson
Assembly-Level Pre-injection Analysis for Improving Fault Injection Efficiency
EDCC-5, Budapest, Hungary, 2005

→ Formal techniques (e.g., symbolic execution?)

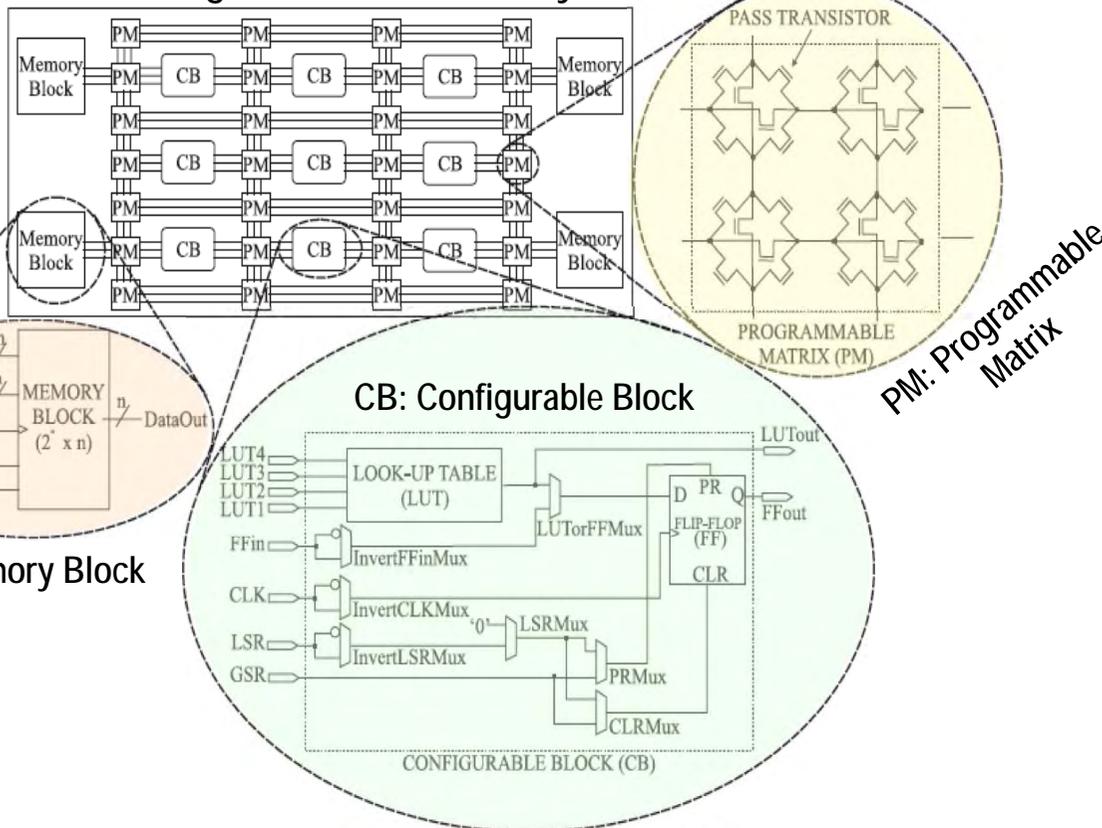
HW-Fault Injection

- Limitation of capabilities of SWIFI techniques wrt HW-level
- Increase of dependability concerns at HW level

- **FPGA-based FI technique**
[De Andrés *et al* 06]

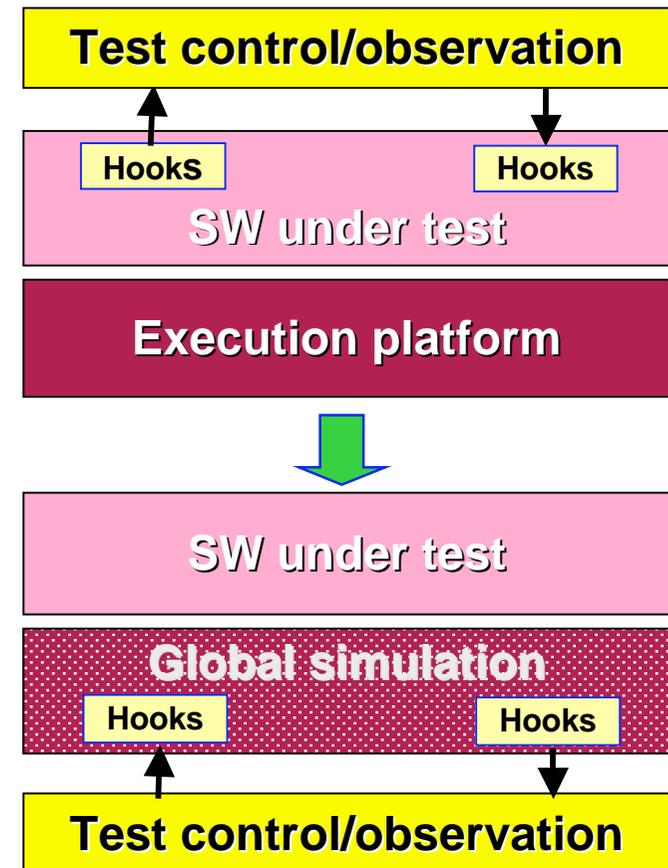
- **Virtual execution platform (incl. proc.)** — ATLAS, F RNTL prog.

Field Programmable Gate Array

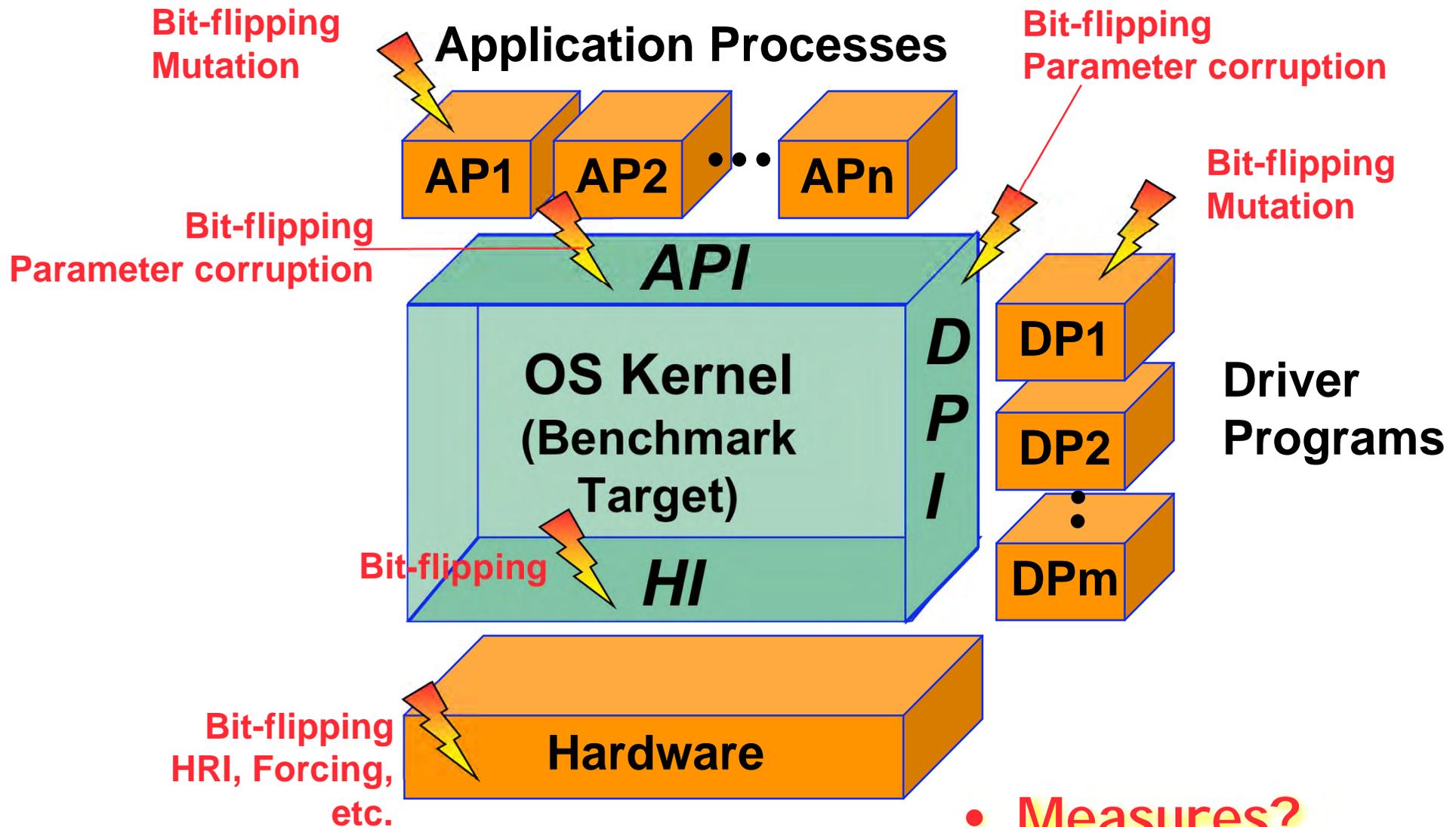


MB: Memory Block

F = stuck-at, open, short, bit-flip, delay, etc.



About Interfaces (SW Executive)



- Measures?
- Representativeness?

Examples of Readouts and Measures

■ Ballista - Failure "scale"

- ◆ **Catastrophic**: crash of the system
- ◆ **Restart**: hang of the system
- ◆ **Abort**: crash of a task
- ◆ **Silent**: no exceptional situation
- ◆ **Hindering**: incorrect error code
- ◆ **Other ?**: Correct error code, non-exceptional tests

■ DBench-OS Measurements

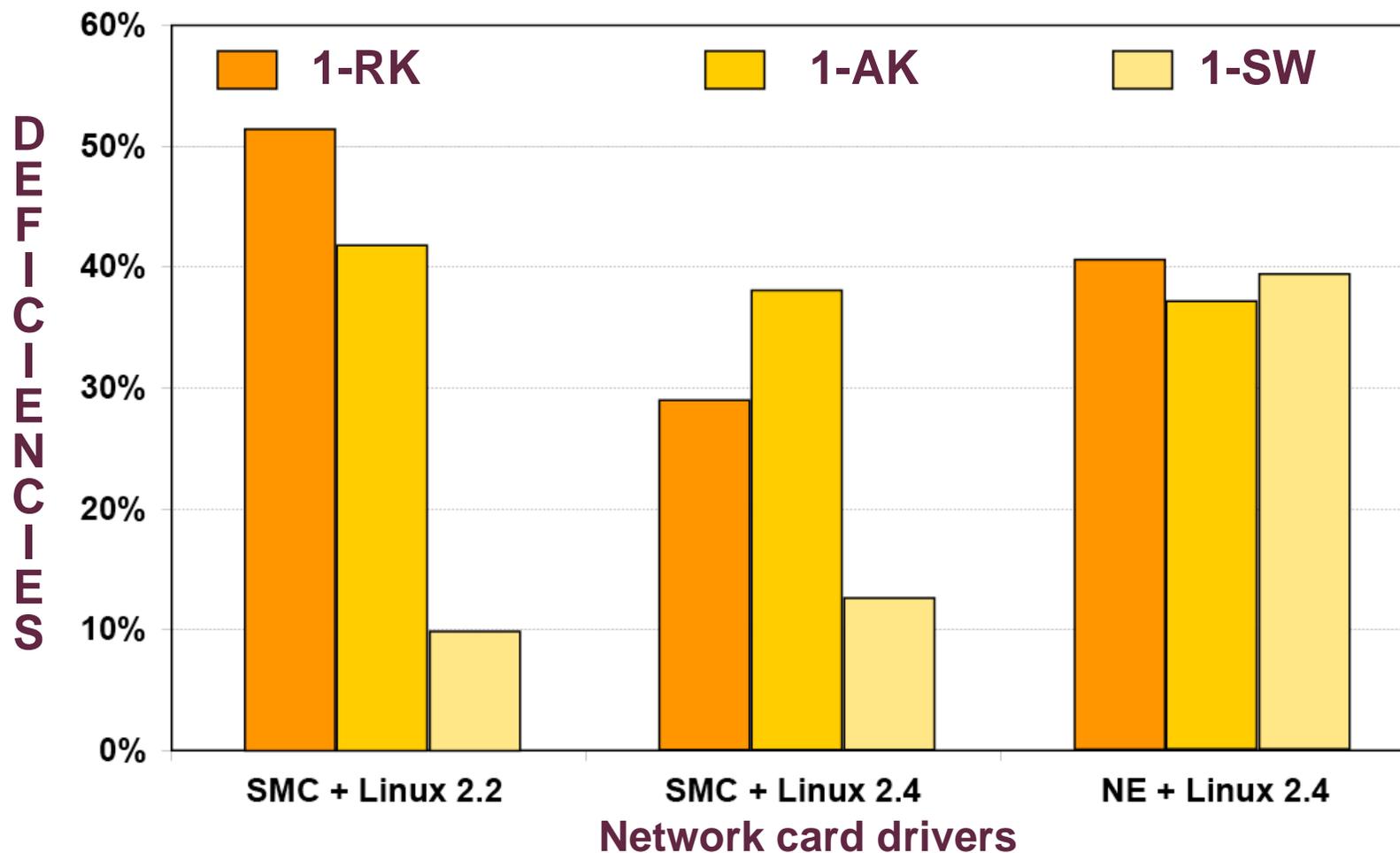
- ◆ **SHd**: system's hang (HW reboot)
- ◆ **SPc**: panic state (SW reboot)
- ◆ **SXp**: exception is raised to applic.
- ◆ **SNS**: no signaling
- ◆ **SEr**: error code returned
- ◆ **Reaction time** to faulted system call
- ◆ **Restart time** of OS after execution

■ MAFALDA & RoCADE

- ◆ **WI**: Workload incorrect
- ◆ **KH**: Kernel hang [API]
- ◆ **WA**: Workload Abort [API]
- ◆ **XC**: Exception raised [API]
- ◆ **No Obs.:** No Observation
- ◆ **EC**: Error Code returned [API/DPI]
- ◆ **WC**: Workload Completion



Measures \leftrightarrow Viewpoints



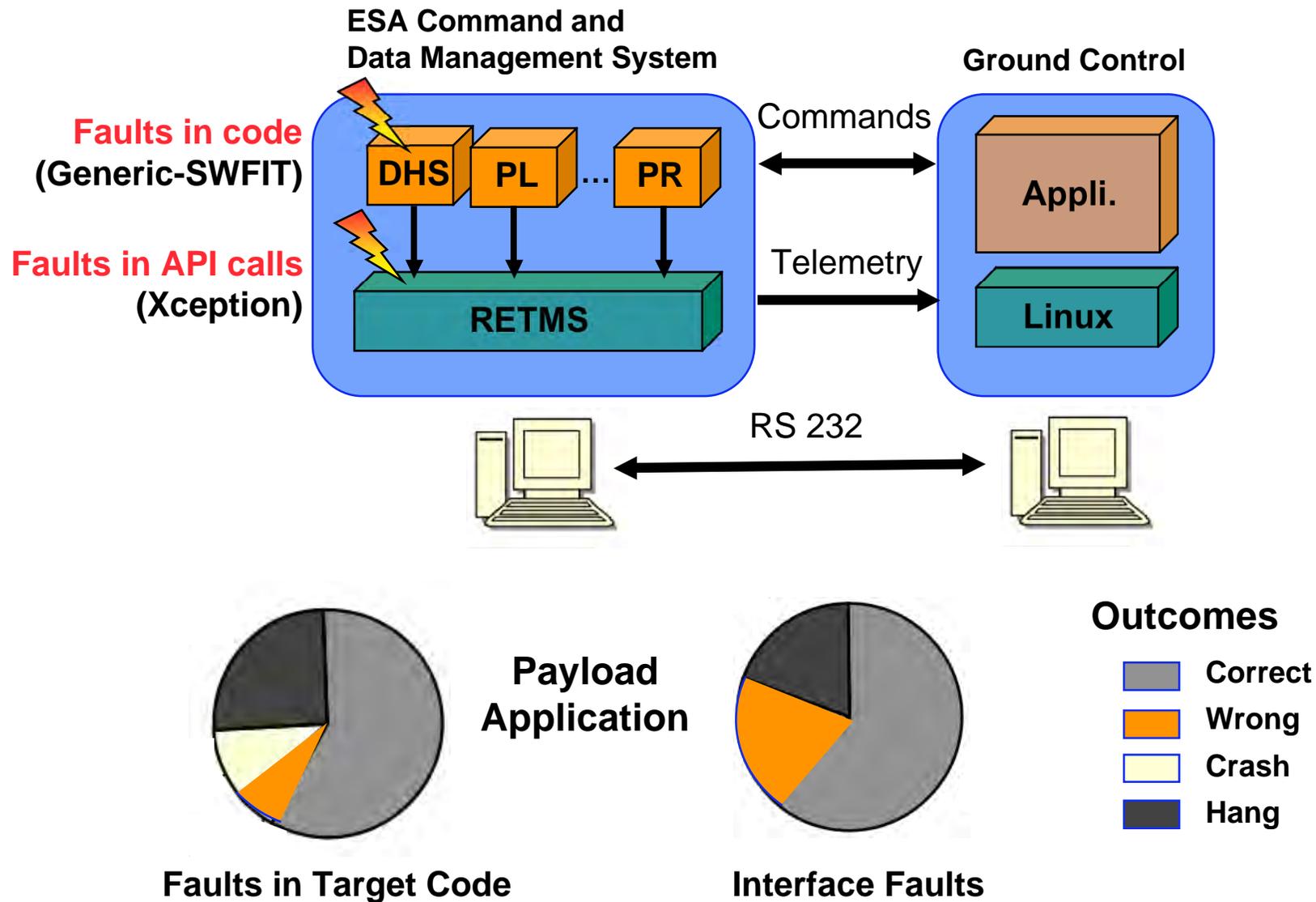
Linux

Kernel call:
parameter
corruption
at DPI

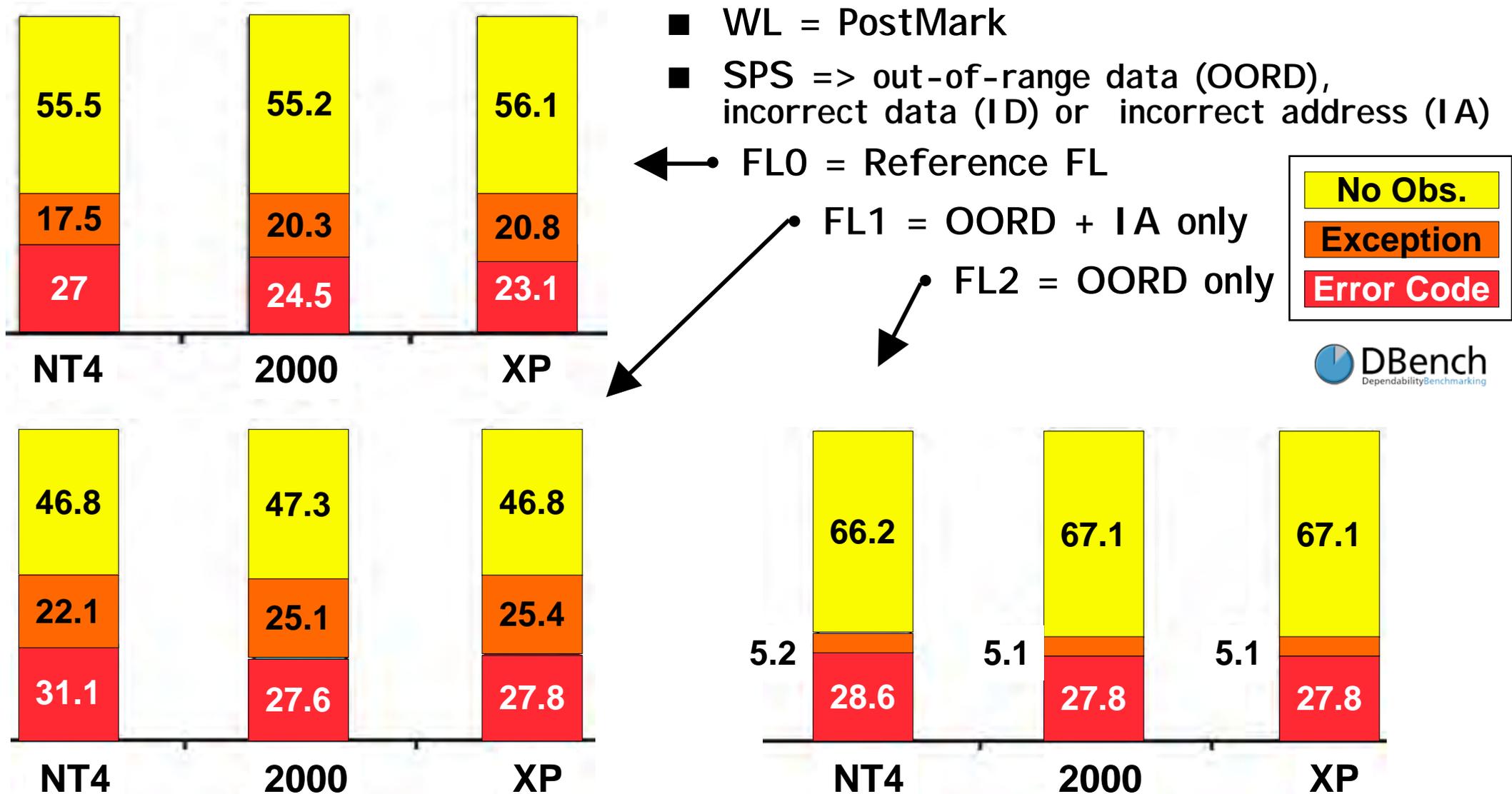


- RK (Responsiveness of the Kernel) = \uparrow error notification
- AK (Availability of the Kernel) = \downarrow kernel hangs;
- SW (Safety of the Workload) = \downarrow delivery of incorrect service

About the Faultload



OS-DBench — API-level Selective Parameter Substitution (Windows Family)

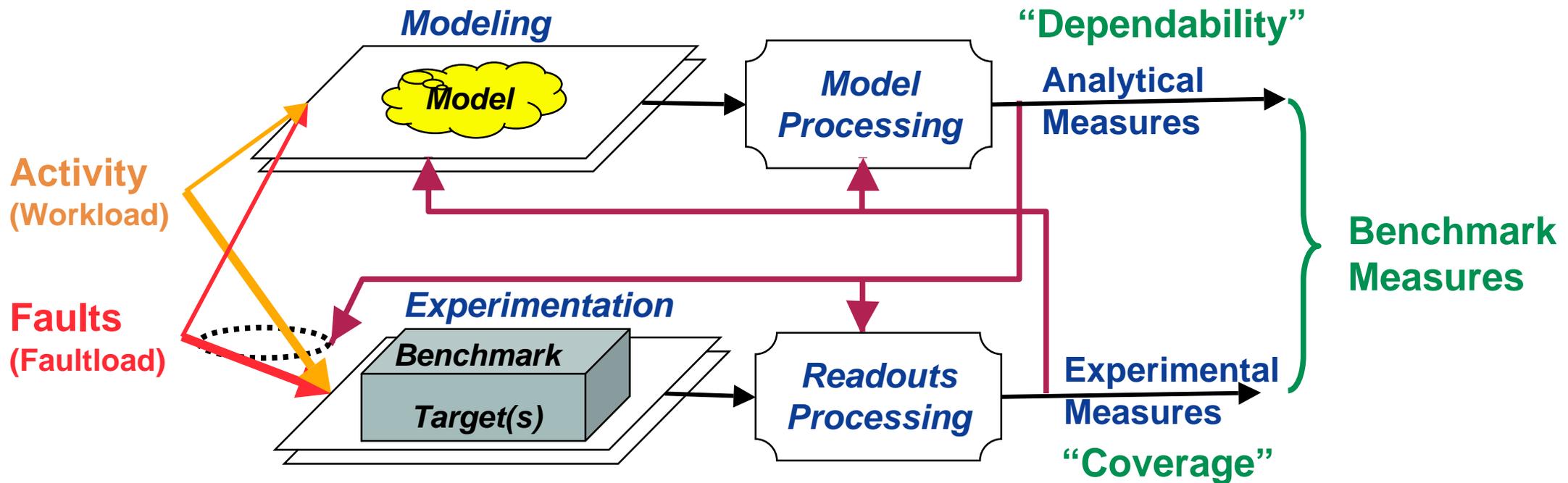


K. Kanoun, Y. Crouzet, A. Kalakech, A. E. Rugina

Windows and Linux Robustness Benchmarks with respect to Application Erroneous Behavior

in *Dependability Benchmarking for Computer Systems* (K. Kanoun and L. Spainhower, Eds.), IEEE CS Press and Wiley, 2008

A Comprehensive Dependability Assessment Frame

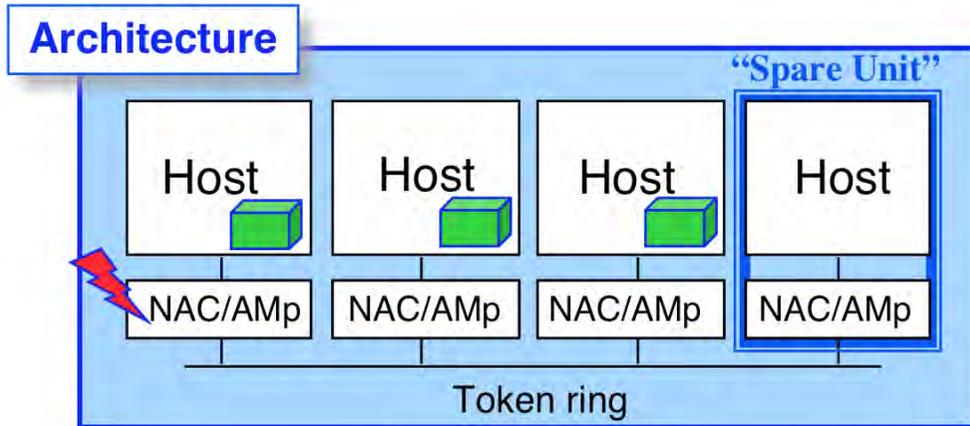


IST Project **DBench** (*Dependability Benchmarking*) — www.laas.fr/DBench and www.dbench.org



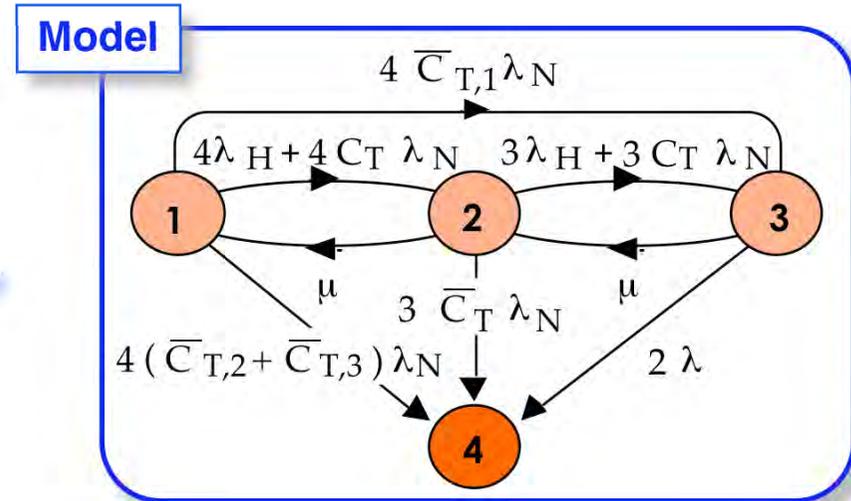
—> Minimal set of data needed from the Target System(s) (architecture, configuration, operation, environment, etc.) to derive actual dependability attributes?

An Early Example: Delta-4 FT Architecture

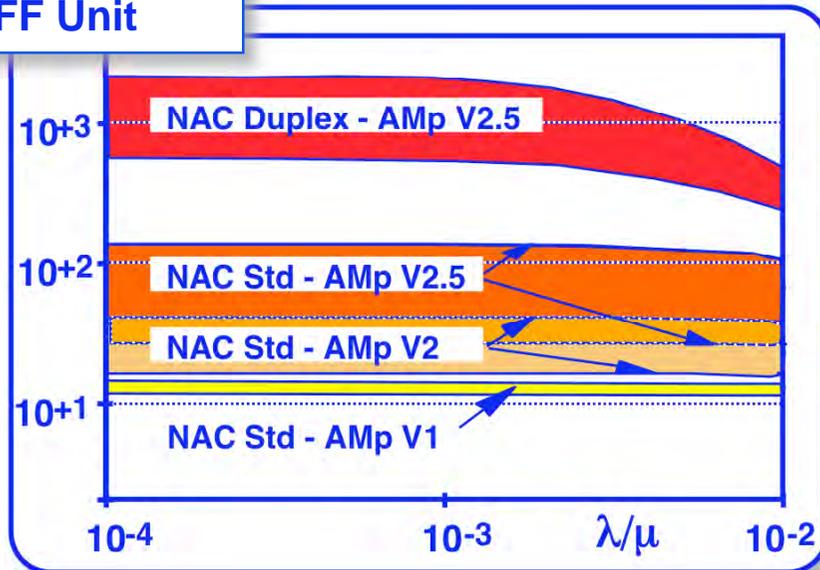


NAC
1) "Standard"
2) Duplex Arch.

AMP
Several releases



MTFF Network
MTFF Unit



Coverage Factors

Target System	C_T	$\bar{C}_{T,1}$	$\bar{C}_{T,2}$	$\bar{C}_{T,3}$
NAC Std - AMP V 1	79,08%	2,32%	11,77%	6,83%
NAC Std - AMP V 2	92,20%	8,73%	2,80%	45%
NAC Std - AMP V 2.5	92,20%	7,79%	1,00%	0,12%
NAC Duplex - AMP V 2.5	99,55%	0,32%	0,00%	0,12%

What about Security Issues?

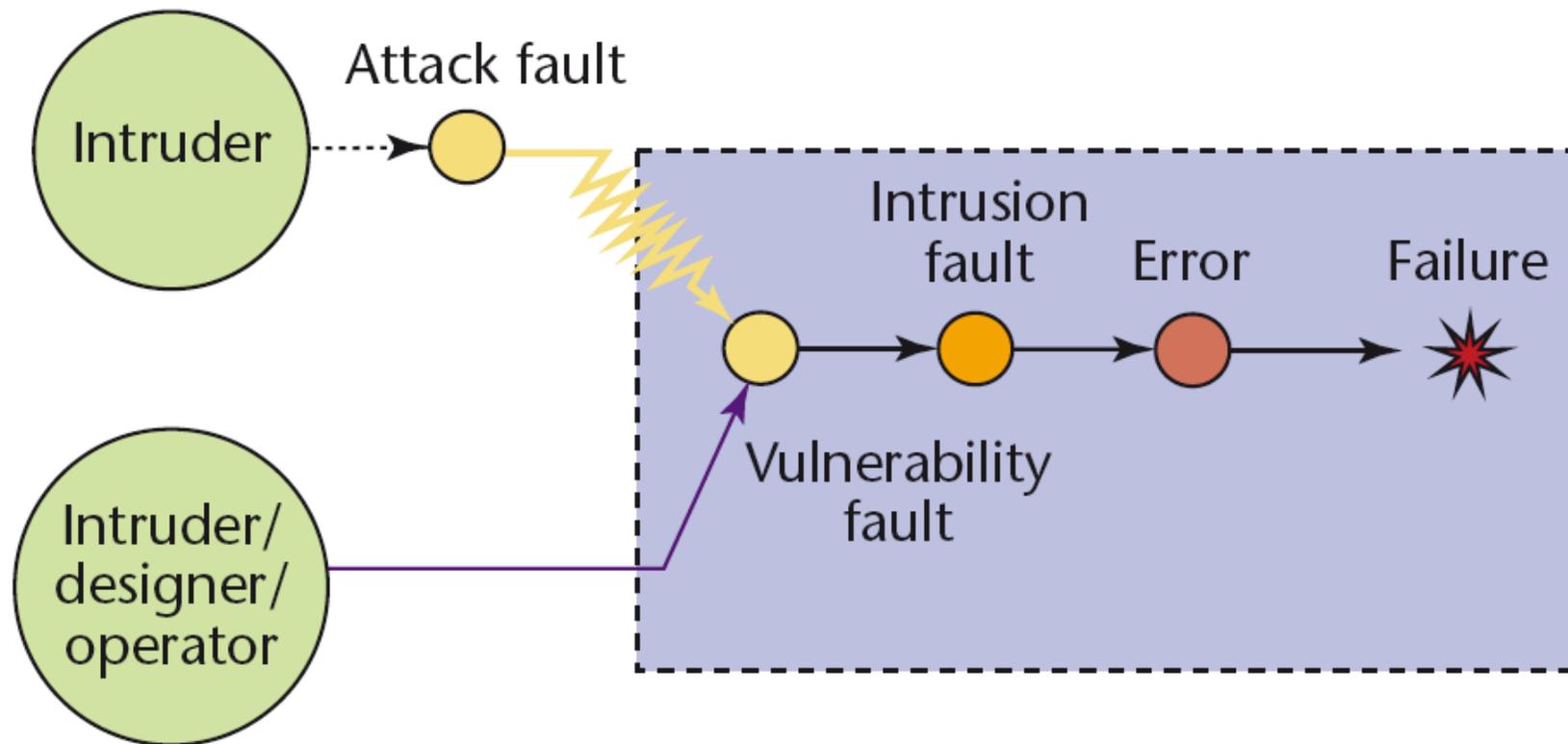
■ Measures

- ◆ What kind of security metrics/measures?
- ◆ Is there an equivalent to the notion of “coverage”?
- ◆ Significance of “false positives” — e.g., Intrusion Detection Systems

■ Faultload

- ◆ Proper set of faults?
HW (bit flip) and/or SW fault injection
- ◆ Successful security breach = combination of attack and vulnerability
—> A (potential) Analogy wrt Verification/Testing:
Error Propagation = Fault + Activity
- ◆ Impact of SW faults on vulnerabilities wrt to specific attacks
- ◆ HW-related issues (e.g., side channel attacks)
- ◆ HW-induced faults are also a concern (Fault Injection targeting cryptographic circuits + Differential Fault Analysis)
- ◆ Built-In-Self-Testing facilities -> Vulnerabilities wrt Security

The MAFTIA Attack/Vulnerability/Intrusion Pathology Model



Looking Ahead

- Significant **conceptual and technological** advances
- **Fault Injection-based assessment**: recognized as a successful technique and is now largely applied in industry
- **Dependability Benchmarking**: rising and promising, but still a lot to do...
- Re-establish powerful and flexible **HW-layer fault injection technologies** (mandatory to test HW-implemented FTMs)
- **Faultload Representativeness**: comprehensive hierarchical fault/error models and related transfer functions
- **Agreed/Shared Benchmarking Frame, Repository & Procedures**
 - ◆ Fairness → common standard interfaces
 - ◆ Experiments → Single fault / run vs. sequence of faults / run
- **Security issues** (Faultload, Metrics/Measures)
- **Mobile and Ubiquitous Computing**

Thanks to...

- Colleagues of the Dependable Computing and Fault Tolerance research group at LAAS-CNRS
- Many partners of Delta-4, PDCS, DeVA & DBench projects, members of IFIP WG 10.4, and of the “FTCS-DSN” community

Road books...

- A. Benso, P. Prinetto (Eds.), **Fault Injection Techniques and Tools for Embedded Systems Reliability Evaluation**, Frontiers in Electronic Testing, #23, 245p., Kluwer Academic Publishers, London, UK, 2003.
- **SIGDeB: IFIP WG 10.4 on Dependable Computing and Fault Tolerance Special Interest Group on Dependability Benchmarking** [www.dependability.org/wg10.4/SIGDeB]
- **DeBench: Dependability Benchmarking Project (IST-2000-25425)** [<http://www.laas.fr/DBench>]
- K. Kanoun, L. Spainhower (Eds.), **Dependability Benchmarking for Computer Systems**, 362p., Wiley-IEEE CS Press, 2008.