OS Dependability Benchmarking (La sûreté de fonctionnement comme critère de choix d'un OS)

Karama Kanoun
Karama.Kanoun@laas.fr

LAAS-CNRS

Research Group on Dependable Computing and Fault Tolerance
Tolérance aux fautes et sûreté de fonctionnement (TSF)

Conférence du 22 octobre 2012, Toulouse

DBench
DependabilityBenchmarking

European Project

Definition of a conceptual framework & experimental environment(s)
for benchmarking the dependability of COTS and COTS-based systems

Outcomes of DBench

- **Partners**

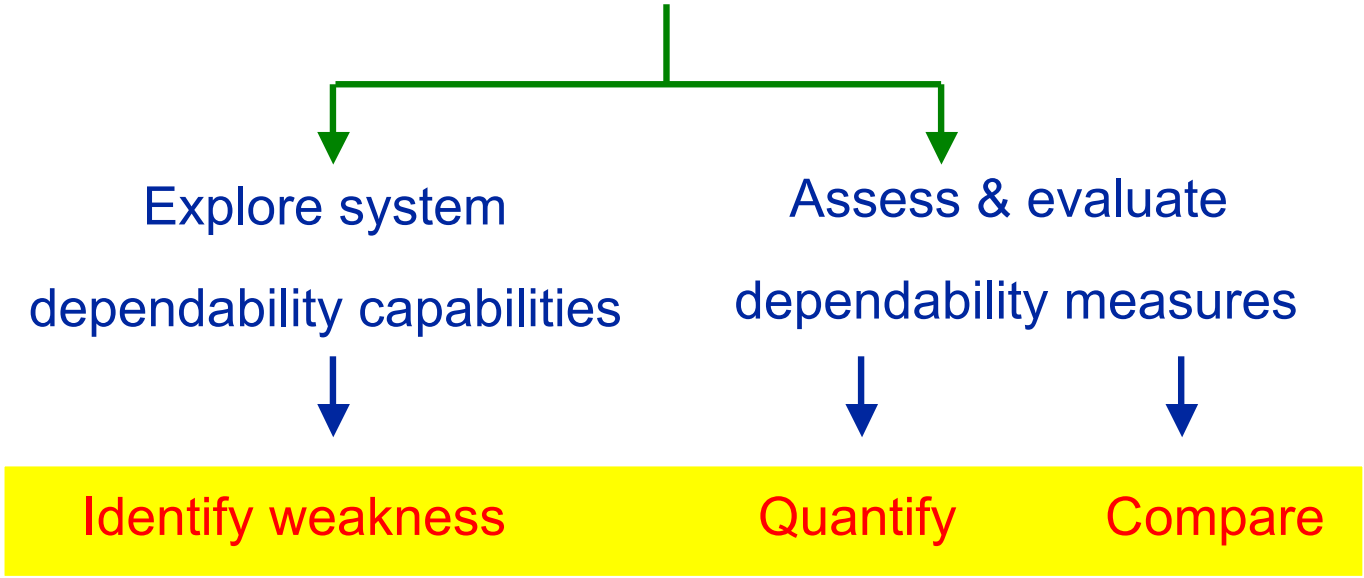
  LAAS-CNRS, France (Coordinator), Critical Software, Portugal, University of Coimbra, Portugal, Friedrich Alexander University, Germany Polytechnic University of Valencia, Spain

- **Advisory Board**

  Astrium (F), CMU (USA), INDRA (E), Oracle (P), Saab Ericsson Space (S), Thales (F)

DBench
DependabilityBenchmarking

Benchmarking Objectives



Benchmarking the dependability of a system consists in evaluating dependability or performance-related measures experimentally or based on experimentation and modeling

→ characterize objectively the system behavior in the presence of faults.

⇒ **non-ambiguous comparison of alternative solutions**

DBench
DependabilityBenchmarking

Benchmarks Developed

- General-purpose operating systems
  - Robustness and timing measures, faulty application, faulty drivers
- Real-Time kernels in onboard space systems
  - Predictability of the kernel response time, faulty application
- Engine control applications in automotive systems
  - Impact of application failures on system safety, transient hardware faults
- On-line transaction processing (OLTP) environments
  - TPC-C-based, operator, software & hardware faults
  - Web-servers, SPEC-based, operator, software & hardware faults

http://www.laas.fr/DBench/

Software Systems Dependability Evaluation

Information on software behavior

- Field data
- Data from development
- Controlled experiments

Dependability benchmarking

Evaluation of dependability measures / features

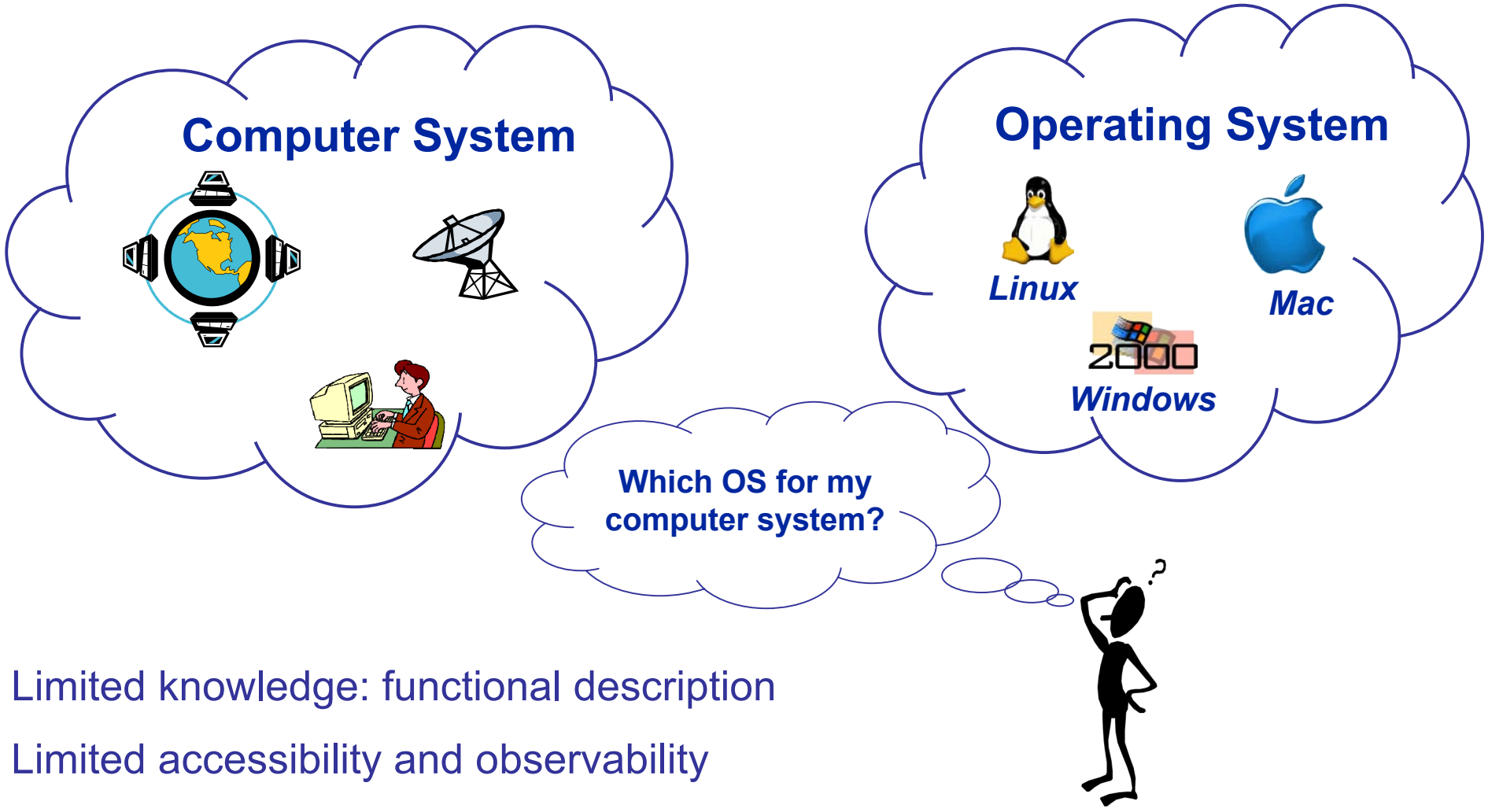
in a non-ambiguous way → comparison

⇓

Properties

Reproducibility, portability, representativeness, acceptable cost

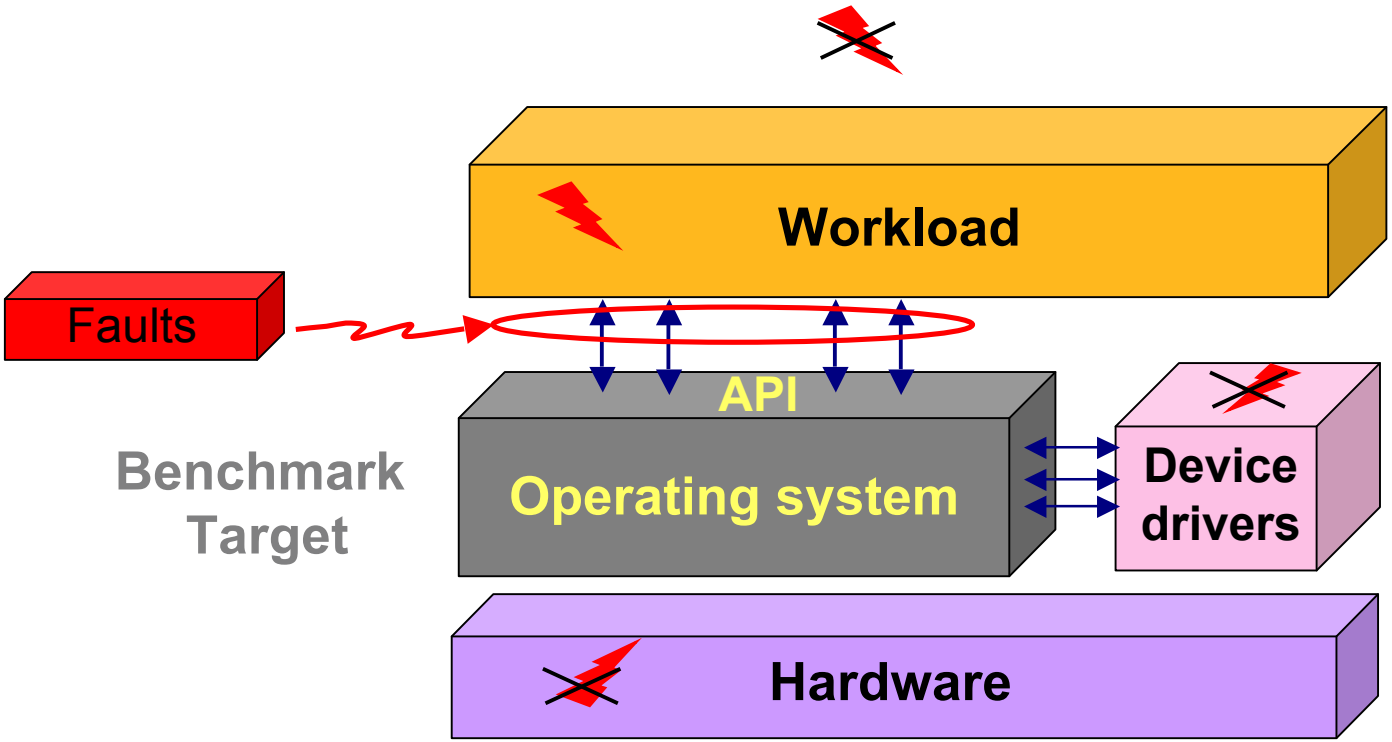
OS Benchmarking: User Point of View



- Limited knowledge: functional description
- Limited accessibility and observability
- Limited intrusiveness and interference

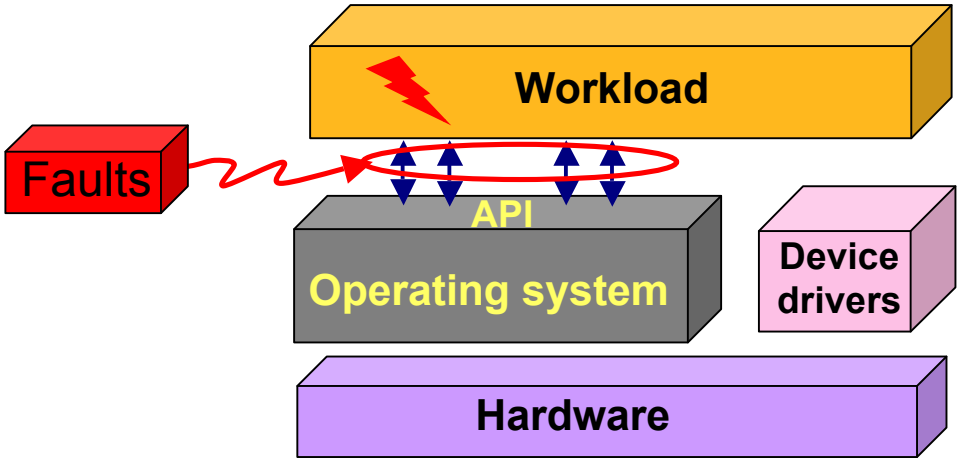
⇒ **Black-box approach ⇒ robustness benchmark**

Benchmarking wrt class of faults?



Wrt application erroneous behavior

OS Benchmark & Measures



OS Outcomes

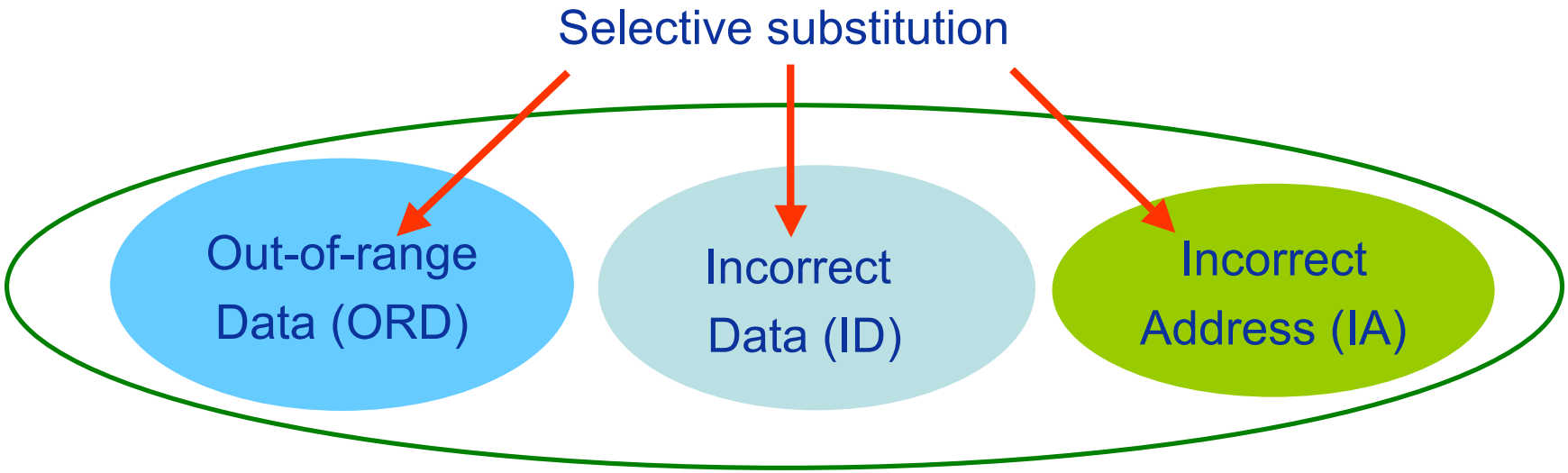
SEr	Error code
SXp	Exception
SPc	Panic
SHg	Hang
SNS	No signaling

Measures

- POS: OS Robustness [%SEr %SXP %SPc %SHg %SNS])
- Texec: OS reaction time in the presence of faults
- Tres: OS Restart time after fault insertion

Execution Profile

- Workload
  - TPC-C Client, Java Virtual Machine, PostMark
- Faultload
  - Corruption of parameters of all system calls

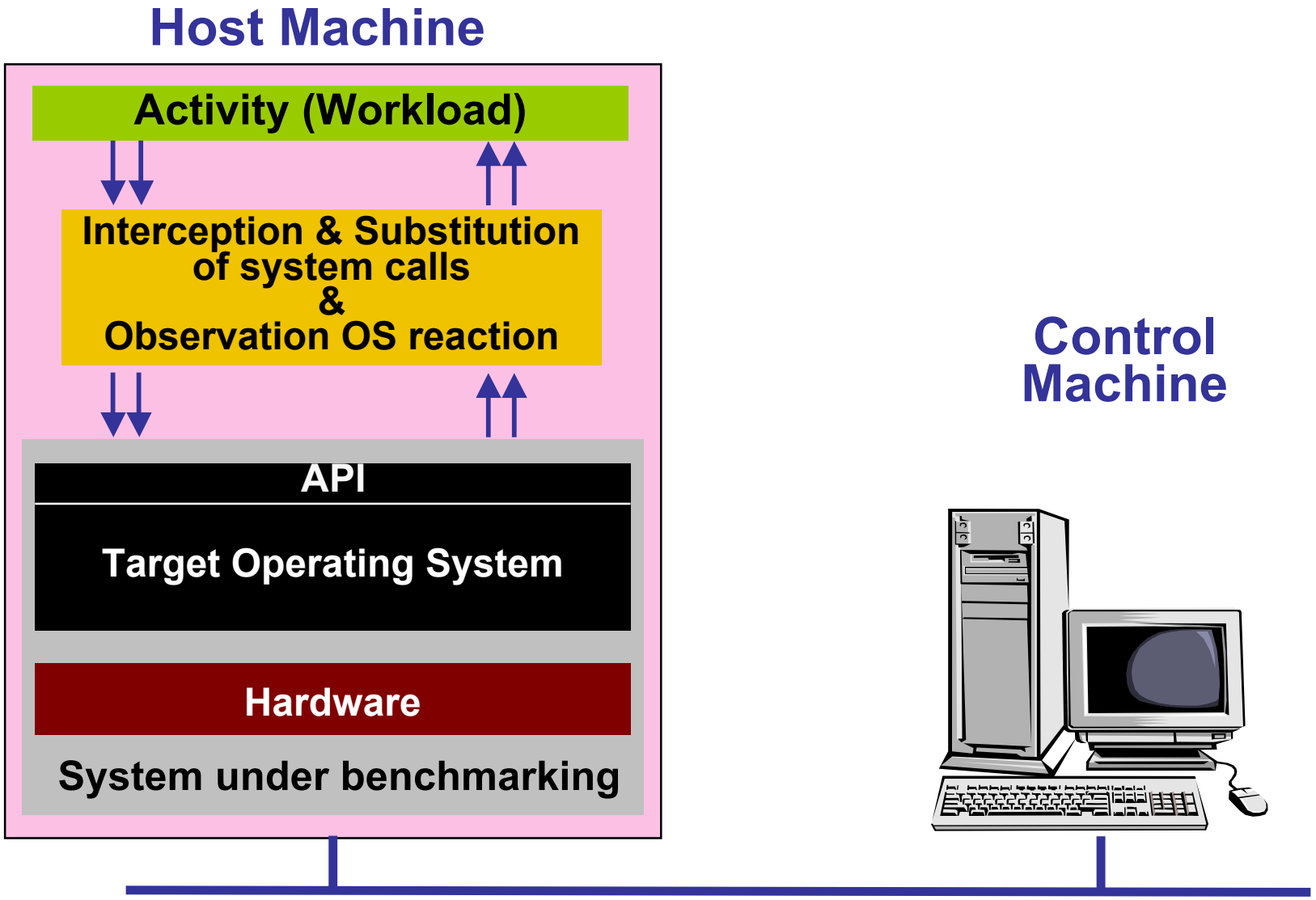


Dependability Benchmarks with PostMark WL

	# system calls	# experiments
Windows NT 4	25	418
Windows 2000	25 + 1 + 1	433
Windows XP	25 + 1	424
Windows NT 4 Server	25	418
Windows 2000 Server	25 + 1 + 1	433
Windows 2003 Server	25 + 1 + 1	433

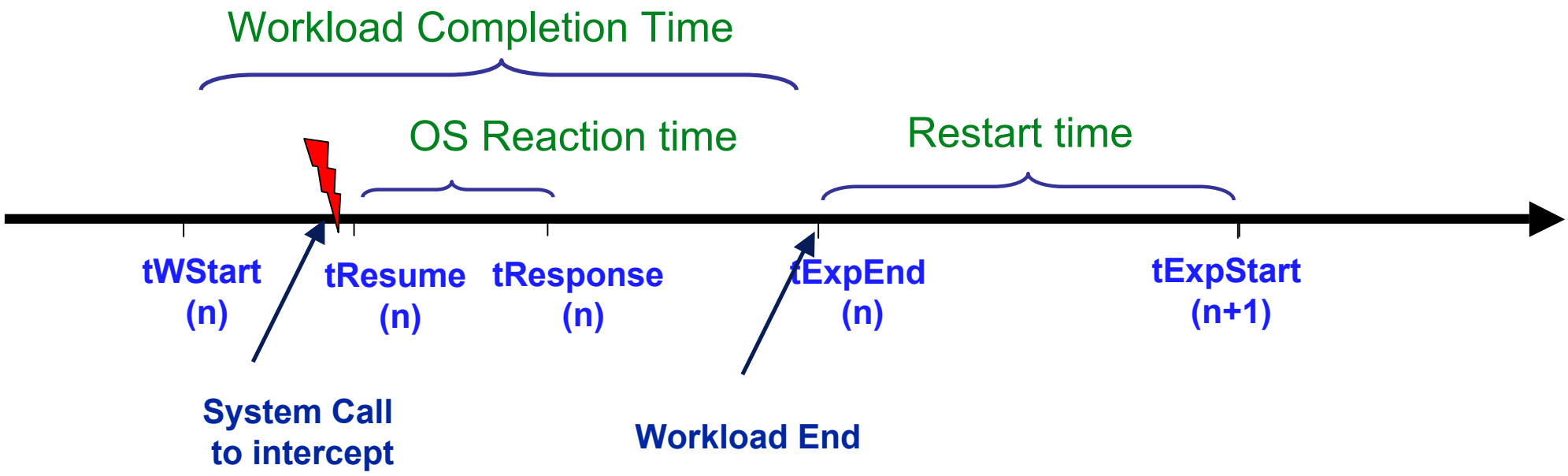
Linux 2.2.26	15 + 1	206
Linux 2.4.5	15 + 1	206
Linux 2.4.26	15 + 1	206
Linux 2.6.6	15 + 2	228

Experimental set-up



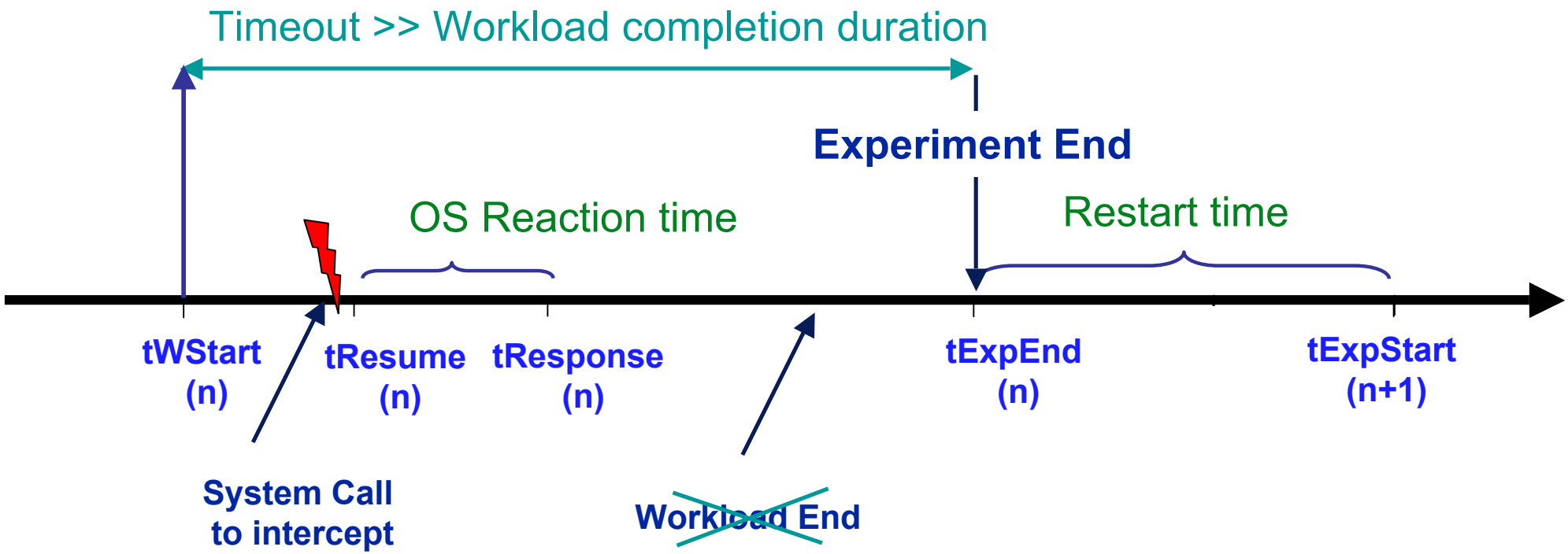
Measurements

Experiments with Workload completion



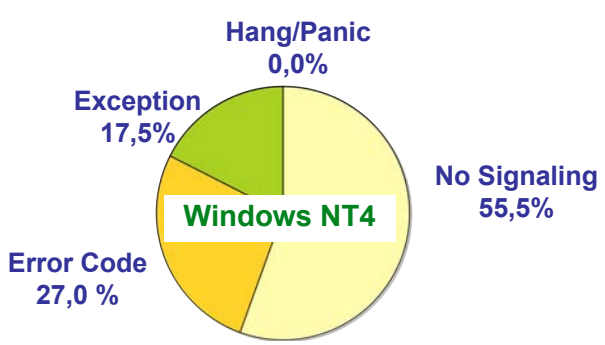
Measurements

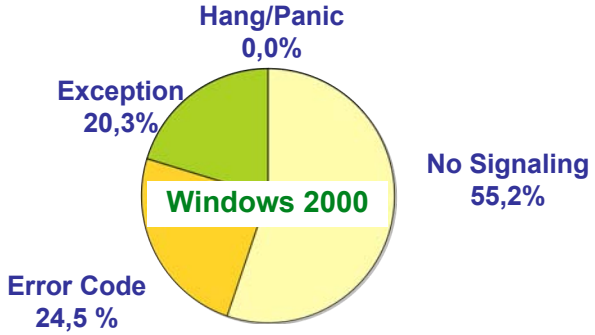
Experiments without Workload completion

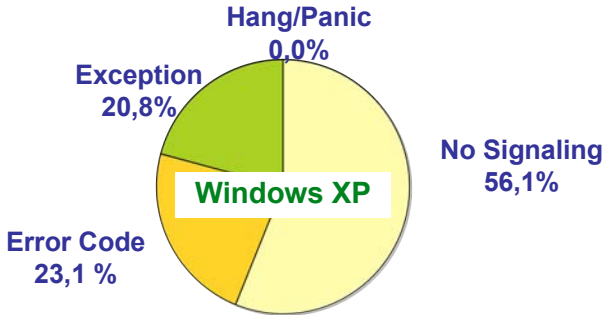


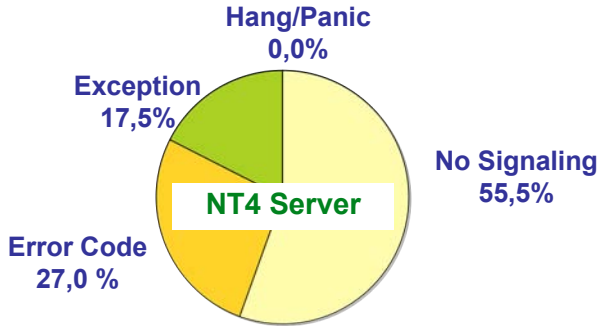
Robustness (WL = PostMark)

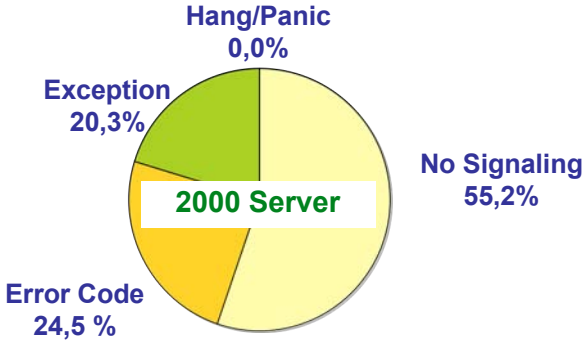
Windows

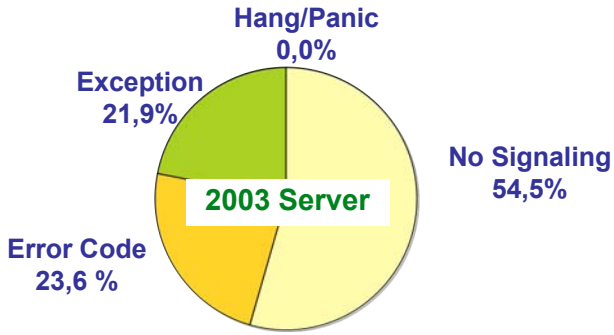




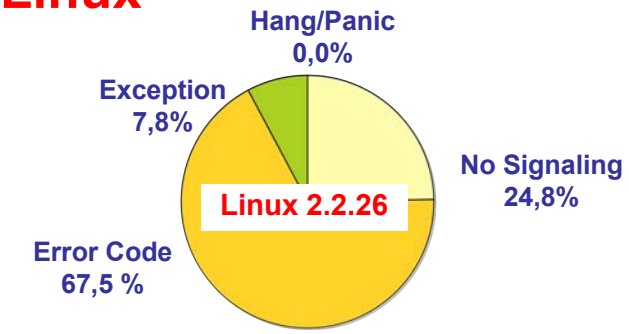


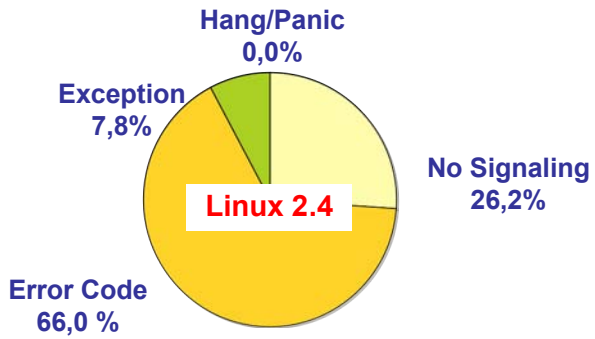


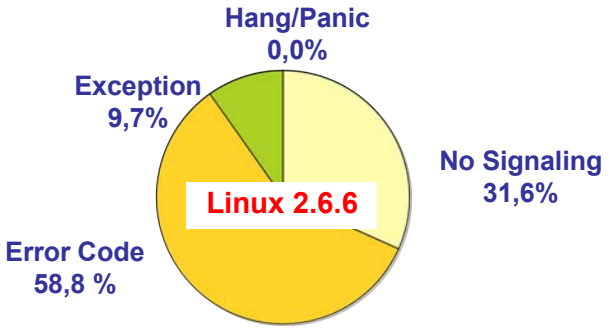




Linux

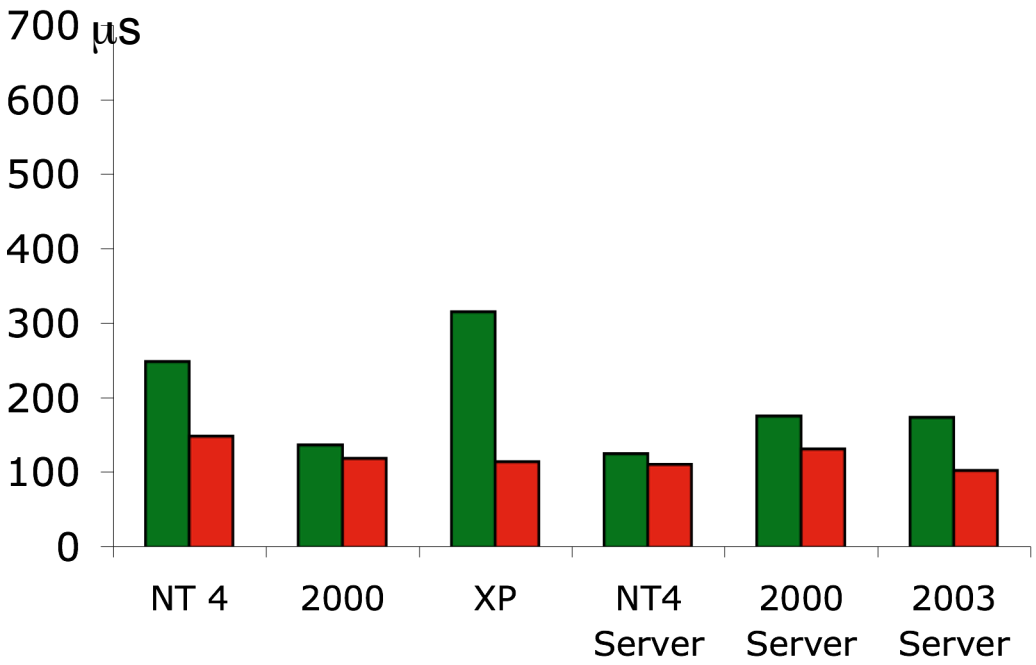




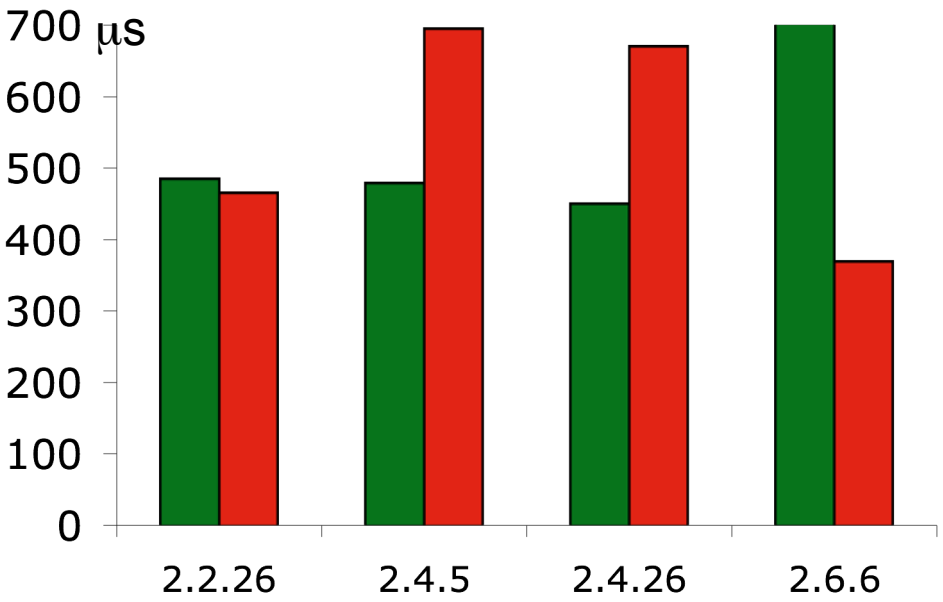


OS Reaction Time (WL = PostMark)

Windows



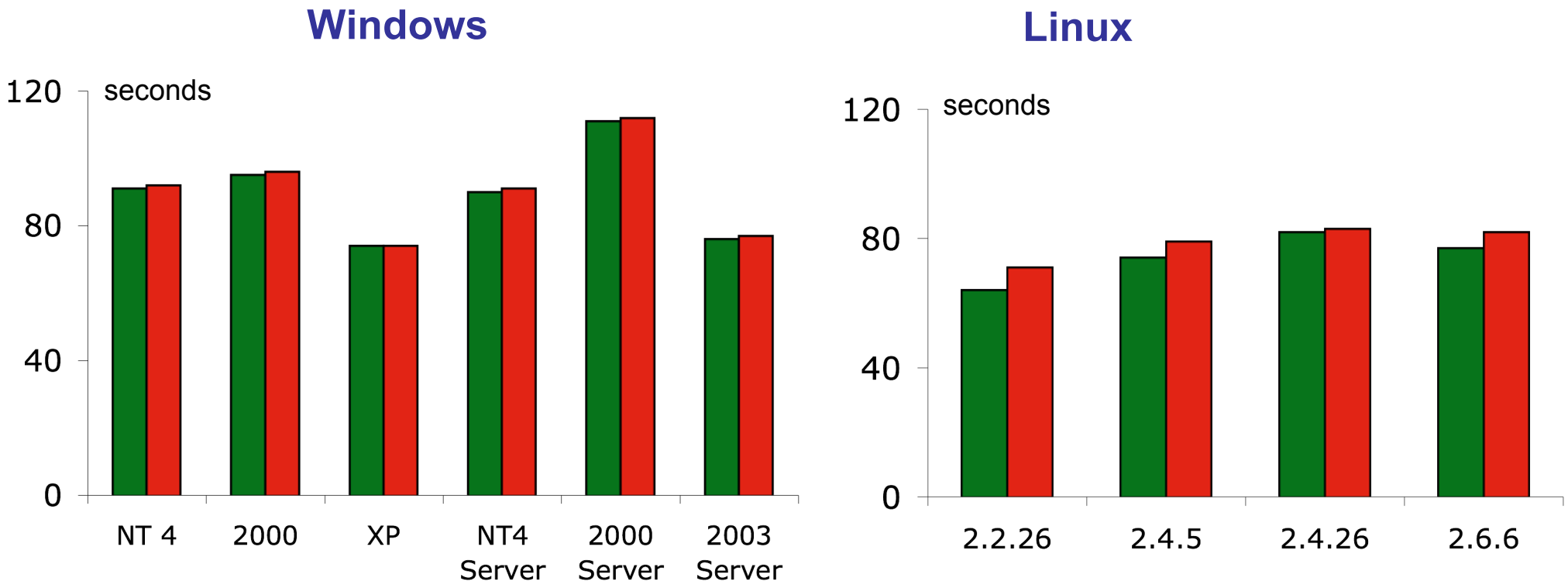
Linux



In the presence of faults

Without parameter corruption

Restart Time (WL = PostMark)

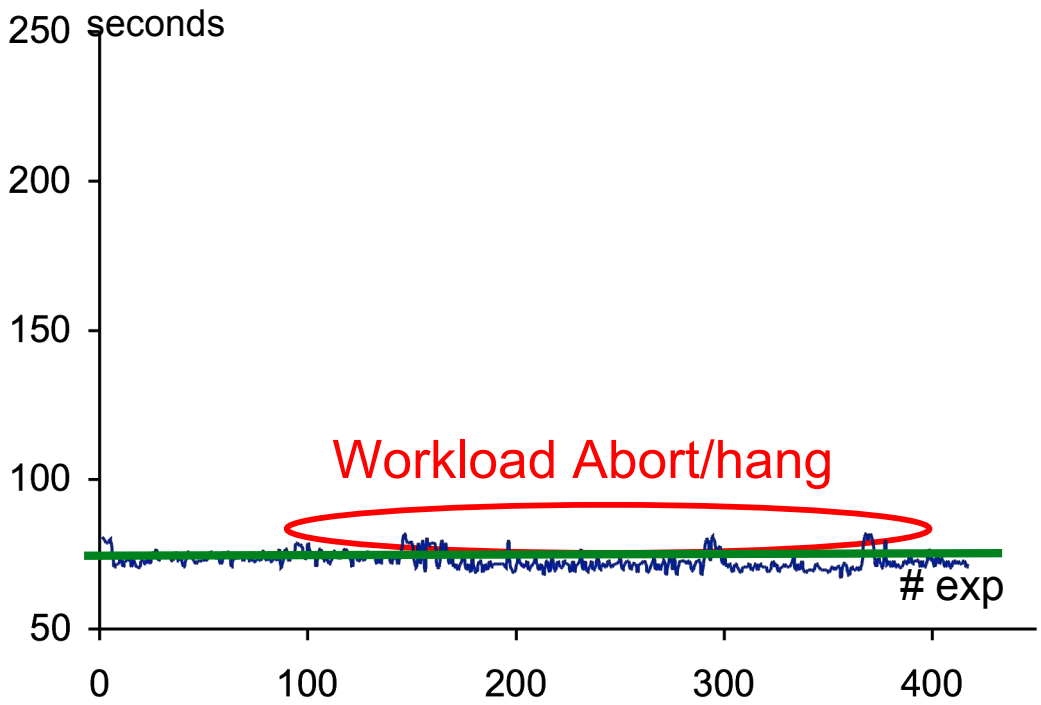


In the presence of faults

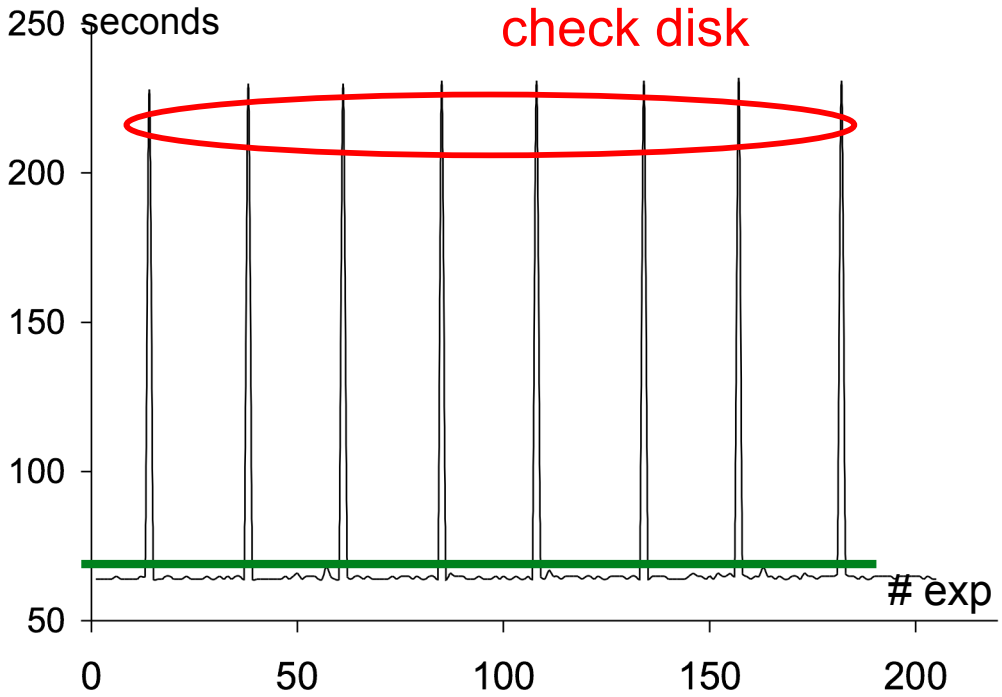
Without parameter corruption

Restart Time (WL = PostMark)

Windows XP



Linux 2.2.26



Validation of properties

- Reproducibility
  - ✓ By construction
  - ✓ Set of faults
    - System Calls to be corrupted
    - Substitution values
- Repeatability
  - ✓ Each benchmark has been executed three times
    - Same robustness
    - Variation of the reaction time (< 4% for TPC-C client)
    - Variation of the restart time (< 3% for TPC-C client)

Validation - Sensitivity Analyses wrt Faultload

	Parameter corruption type			# experiments	
	Incorrect Data	Incorrect Address	Out-of-range Data	Windows NT4	Linux
Faultload 0	x	x	x	418	206
Faultload 1		x	x	331	135
Faultload 2			x	77	55

- Equivalence of versions of the same family
- Same comparison results between the two families

Additional analysis: incorrect data = out-of-range data in the context

Validation - Cost/effort

- Benchmark implementation duration

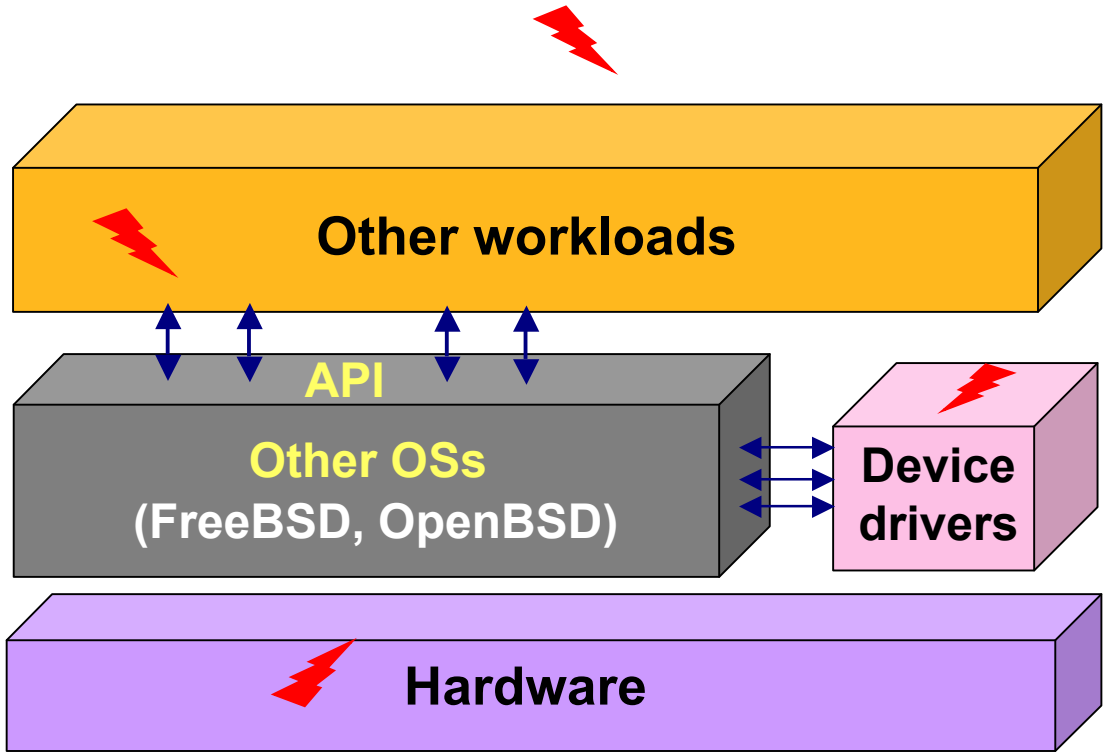
  For each OS family

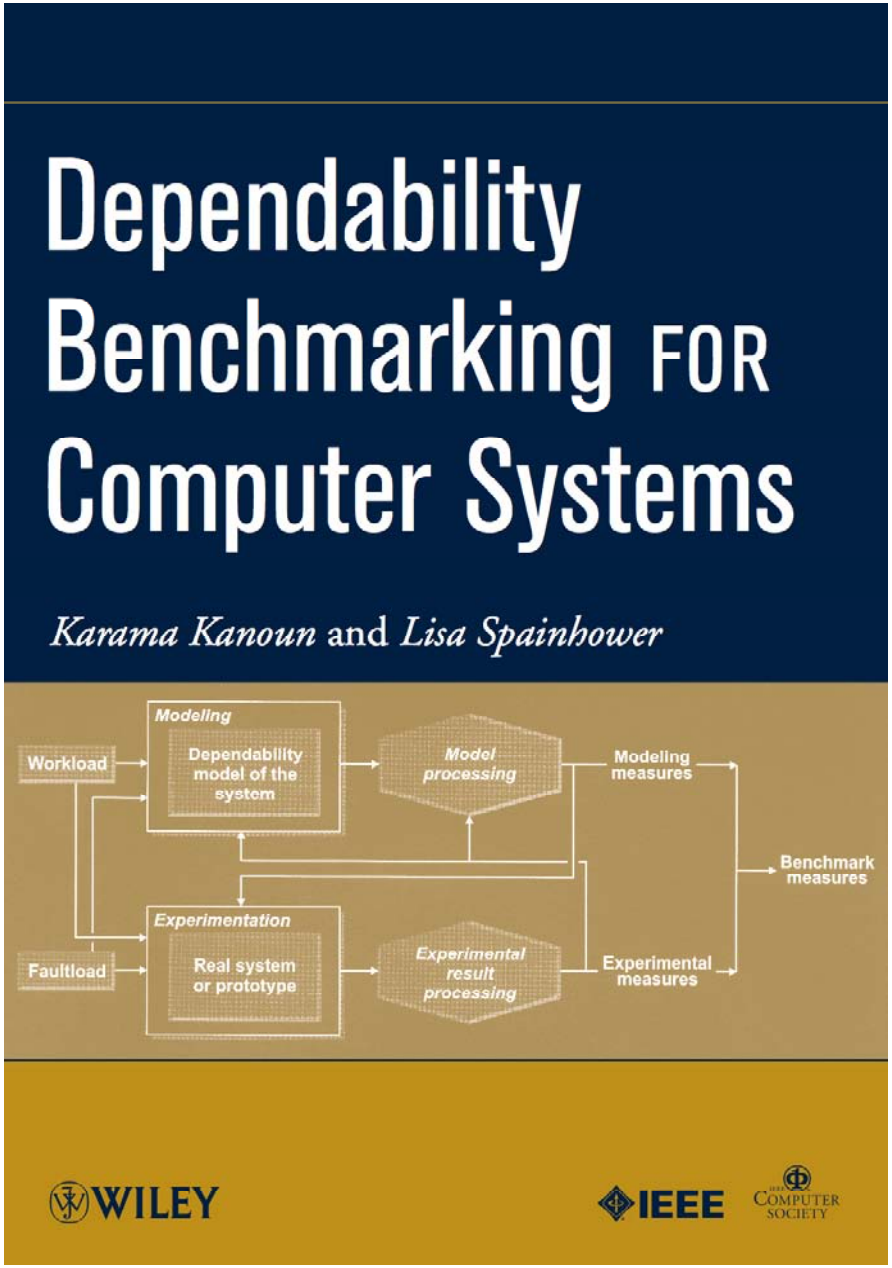
  - ✓ Implementation of the workload: 1 - 3 days
  - ✓ Controller, parameter corruption and observation: 2 weeks
  - ✓ Definition and implementation of fault set: 1 week

- Experiment duration

	Windows	Linux
TPC-C client	2 days	1 day
Postmark	2 days	1 day
JVM	4 days	2 days

Other Benchmarks




Dependability Benchmarking FOR Computer Systems
Karama Kanoun and Lisa Spainhower
Modeling
Workload
Dependability model of the system
Model processing
Modeling measures
Benchmark measures
Experimentation
Faultload
Real system or prototype
Experimental result processing
Experimental measures
WILEY
IEEE
COMPUTER SOCIETY