technische universität dortmund

Proactive memory error detection for the Linux kernel

Diplomarbeit Jens Neuhalfen / TU-Dortmund

Jens Neuhalfen <jens.neuhalfen@cs.tu-dortmund.de>

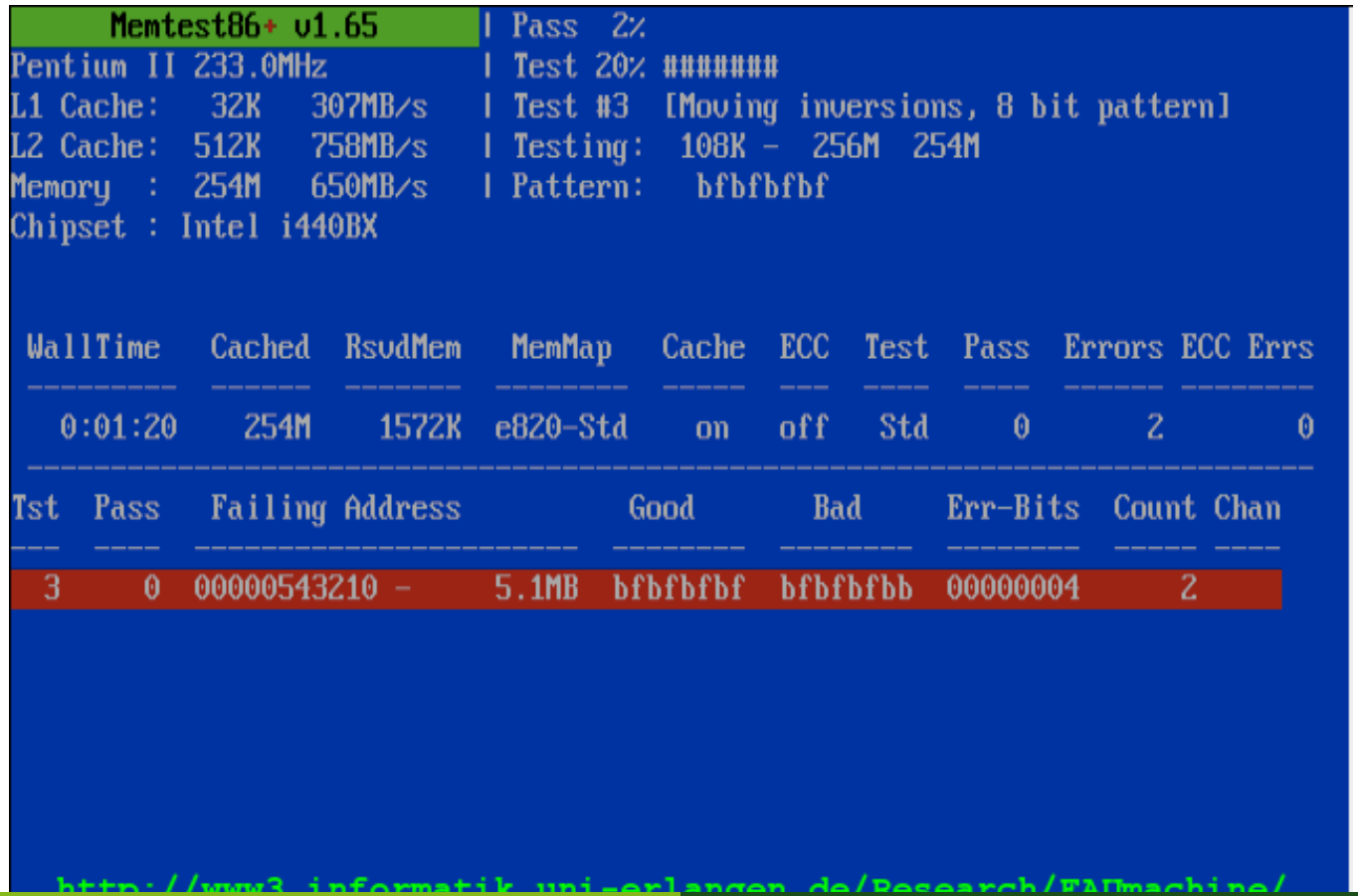
2010-08-02

technische universität dortmund

Inhaltsverzeichnis

technische universität dortmund

Speichertests / Online Speichertests



Was ist Speicher?

technische universität dortmund

Def. nach Hayes

... *memory* is a set of r addressable binary storage cells $M_r := \{C_0, C_1, \ldots, C_{r-1}\}$, where i denotes the address of C_i.
Operations *READ* and *WRITE* can be performed on M_r, and each cell can be written to or read from independently of previous *READ* or *WRITE* operations: M_r is a *random-access memory*.
$Y_{0..n-1}$ denotes the possible states a memory M_r can be in. A single state Y_j is described by vector $(y_{j,0}, y_{j,1}, \ldots, y_{j,r-1})$, with $y_{j,0..r-1} \in \{0,1\}$ being the content of C_i.
The operations *READ* and *WRITE* 0/ *WRITE* 1 are associated with the functions W_i, $\bar{W}_i$ and R_i on the states of M_r with

$$W_i(y_{j,0}, y_{j,1}, \ldots, y_{j,\mathbf{i}}, \ldots y_{j,r-1}) = (y_{j,0}, y_{j,1}, \ldots, 1, \ldots y_{j,r-1}) \quad (1)$$

$$\bar{W}_i(y_{j,0}, y_{j,1}, \ldots, y_{j,\mathbf{i}}, \ldots y_{j,r-1}) = (y_{j,0}, y_{j,1}, \ldots, 0, \ldots y_{j,r-1}) \quad (2)$$

$$R_i(Y_j) = Y_{j,i} \quad (3)$$

Quelle: [65] Hayes, J. P.: Detection of Pattern-Sensitive Faults in Random-Access Memories. In: IEEE Trans. Comput. 24 (1975), Nr. 2, S. 150–157.

Was ist Speicher II?

technische universität dortmund

Def. nach Hayes

$$z(X_i, Y_j) = \begin{cases} y_{j,i} & \text{if } X_i = R_i \\ — & \text{if } X_i = \bar{W}_i \end{cases} \tag{4}$$

The functions $F_0 = \{W_i, \bar{W}_i, R_i, z\}$ with $i \in \{0 \ldots r-1\}$ completely describe the behaviour of M_r. A *fault* is said to occur, when the functions $F_0 = \{W_i, \bar{W}_i, R_i, z\}$ change to $F = \left\{ W_i^F, \bar{W}_i^F, R_i^F, z^F \right\}$ where $W_i^F, \bar{W}_i^F, R_i^F$ with $i = 0, 1, \cdots r-1$ are arbitrary mappings on $\{0,1\}^r$, and z^F is any mapping $\{0,1\}^r \rightarrow \{0,1\}$.

Quelle: [65] Hayes, J. P.: Detection of Pattern-Sensitive Faults in Random-Access Memories. In: IEEE Trans. Comput. 24 (1975), Nr. 2, S. 150–157.

Faultmodels: Was sind Speicherfehler?

technische universität dortmund

„Wenn sich der (logische) Inhalt des Arbeitsspeichers ohne zutun eines Programms ändert.". D.h. wurde in eine Speicherzelle zuletzt der Wert a geschrieben und es wird der Wert a' mit $a' \neq a$ gelesen, dann war das ein (Speicher)fehler.

Fehlermodell nach Hayes (stark vereinfacht)

Der Wert, der aus einer Speicherzelle ausgelesen wird, kann vom Inhalt anderer Zellen abhängen.
Testkomplexität unter der Einschränkung von q überlappenden Nachbarschaften mit je max. einem Fehler (*SPSF*): $O((4q+3) * 2^q * r)$

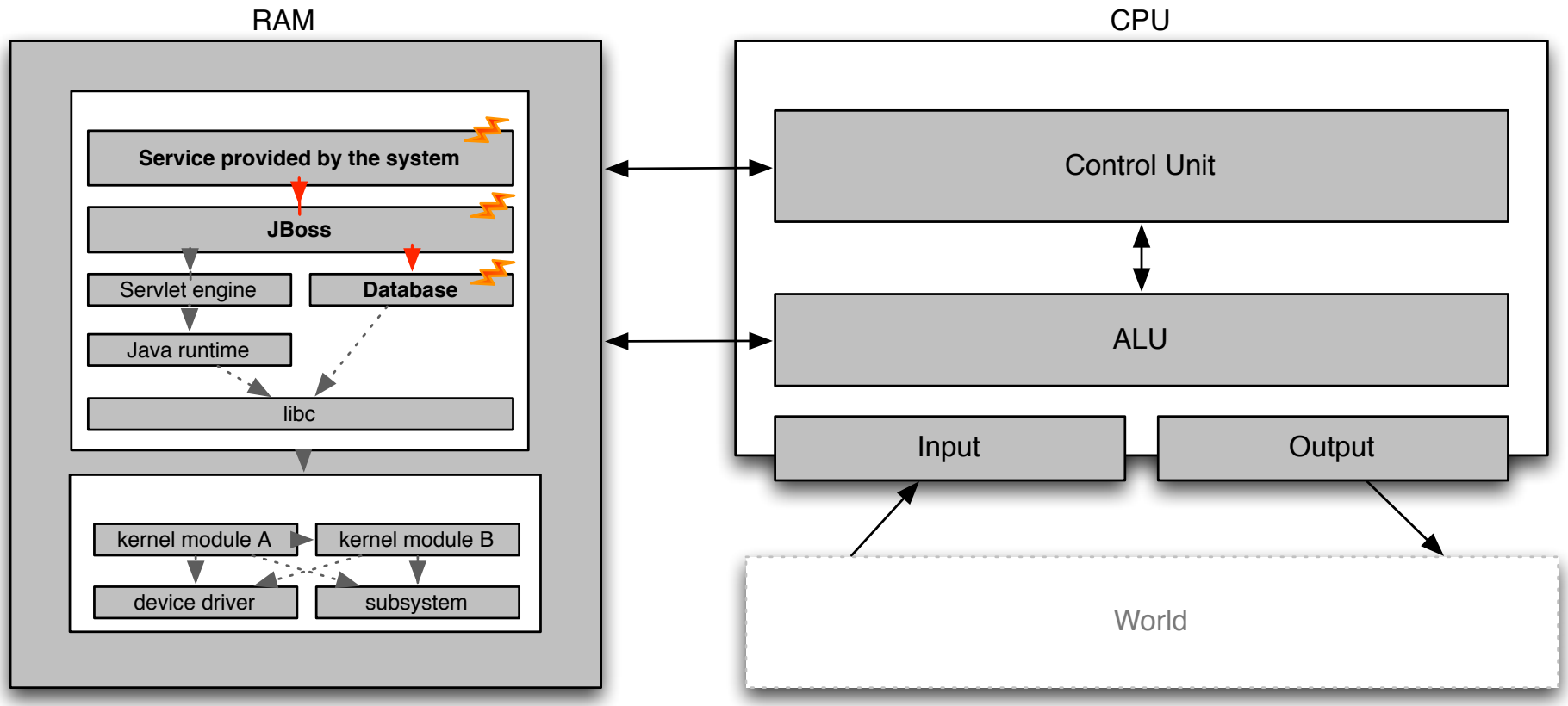
Fehlermodell nach Nair (stark vereinfacht)

Bestimmte Hardwarefehler können den Wert, der aus einer Speicherzelle gelesen wird beeinflussen.
Testkomplexität: $O(r)$

[95] Nair, R. ; Thatte, S.M. ; Abraham, J.A.: Efficient Algorithms for Testing Semiconductor Random-Access Memories. In: Computers, IEEE Transactions on C-27 (1978)

Contracts

technische universität dortmund



DRAM Errors in the Wild

technische universität dortmund

DRAM Errors in the Wild

Studie von Google über $2\frac{1}{2}$ Jahre. Vorgestellt im **Juni 2009** auf der *SIGMETRICS/Performance* [1].

- Mehr als 8% aller Module haben mind. einen Fehler/Jahr.
- ca.32% der Rechner erlebten mind. einen Fehler/Jahr.
- 20% der Maschinen verursachen 90% der Fehler.
- Die Anzahl der Fehler nimmt mit der Zeit signifikant zu.
- $65 - 80\%$ der Rechner mit UE[1] erlebten einen CE im selben Monat.
- Die Auslastung des Rechners hat einen signifikanten Einfluss auf die Fehlerrate.

[1] *CE*: Correctable Errors, *UE* Uncorrectable Errors

Was ist das Problem mit Speicherfehlern?

technische universität dortmund

Wirtschaftliche Gründe:

- **1.) Verfügbarkeit**: Systemabstürze kosten Geld.
- **2.) Persistierung der Daten**: Korrupte Dokumente werden gespeichert.
- **3.) Getroffene Entscheidungen / Pandoras Büchse**

Schlussfolgerung: Speicherfehler treten auf und können schwerwiegende Auswirkungen haben.

Was ist das Problem mit Offline Tests?

technische universität dortmund

Wirtschaftliche Gründe:

- **1.) Verfügbarkeit**: 8 h Speichertest / Jahr $\simeq 99.9\%$ Verfügbarkeit **maximal**.
- **2.) Personeller Aufwand**: Halbjährliche Tests verursachen Personellen Mehraufwand
- **3.) Zwischen den Tests** werden Fehler nicht erkannt.

Schlussfolgerung: Offline Speichertests weisen für den Praxis-Einsatz gravierende Mängel auf.

Zwischenfazit

technische universität dortmund

Zwischenfazit:

- Speicherfehler sind auf verschiedenen Ebenen ein nicht nur wirtschaftliches Risiko.
- ECC Speicher kann dieses Risiko senken.
- Eine Überwachung von (korrigierbaren) Fehlern kann das Risiko weiter senken,
- *wenn* frühzeitig auf Warnsignale (steigende Fehlerraten) geachtet wird.
- Rechner ohne ECC Speicher können CEs *nicht erkennen!*
- Online Speichertests können hier auch/besonders helfen.

Ziel der Arbeit

„How could a feasible software based memory test for low- to midrange servers be designed?”

Fault Management

technische universität dortmund

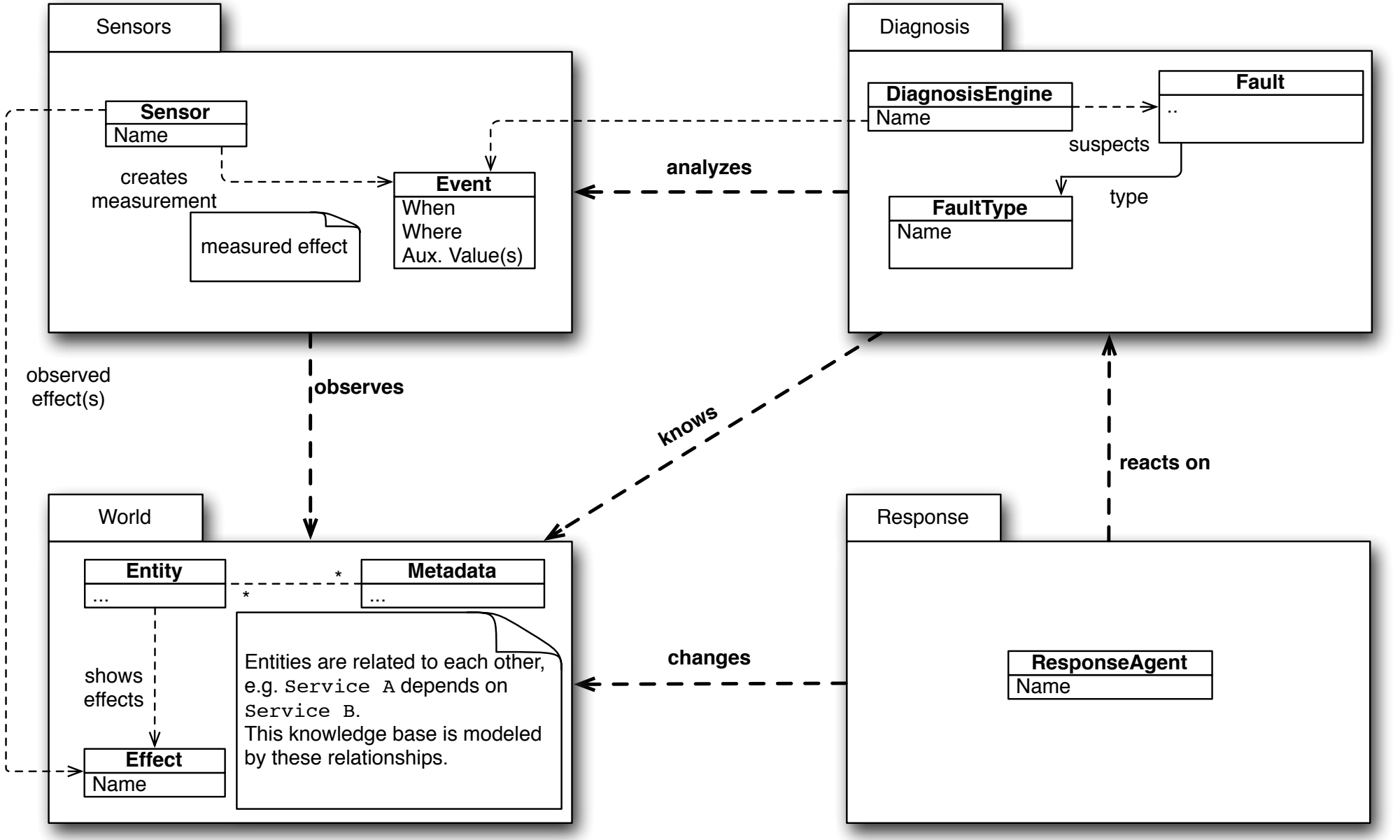
Fault Management

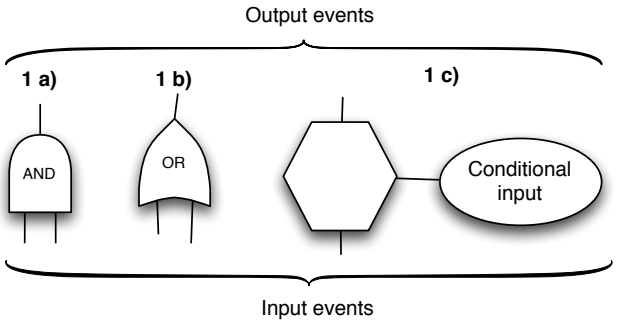
Was ist *Fault Management*?

technische universität dortmund

. . . fault management is the set of functions that **detect, isolate, and correct malfunctions** *. . . include maintaining and examining error logs, accepting and acting on error detection notifications, tracing and identifying faults,* **carrying out sequences of diagnostics tests, correcting faults**, *reporting error conditions . . .*[2]

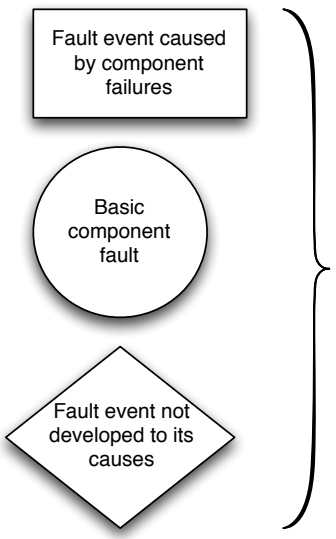
[2] http://en.wikipedia.org/wiki/Fault_management


Sensors
Sensor
Name
creates
measurement
measured effect
Event
When
Where
Aux. Value(s)
Diagnosis
DiagnosisEngine
Name
Fault
..
suspects
type
FaultType
Name
analyzes
observed
effect(s)
observes
knows
reacts on
World
Entity
...
Metadata
...
*
*
shows
effects
Effect
Name
Entities are related to each other,
e.g. Service A depends on
Service B.
This knowledge base is modeled
by these relationships.
changes
Response
ResponseAgent
Name



1 a+b) Gates that denote basic AND / OR funcationallities.

1 c) "Inhibit gate" that is a special AND-type of gatter with a *condition* as input.

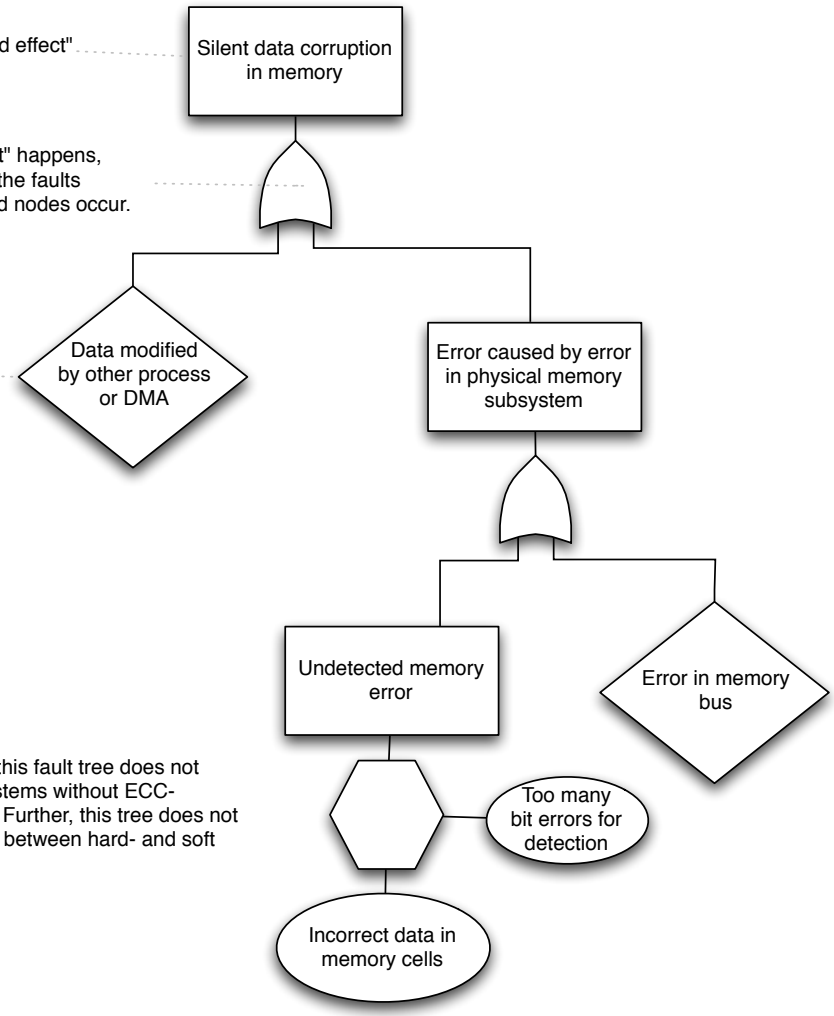


2 a-c) Nodes that describe certain types of faults.

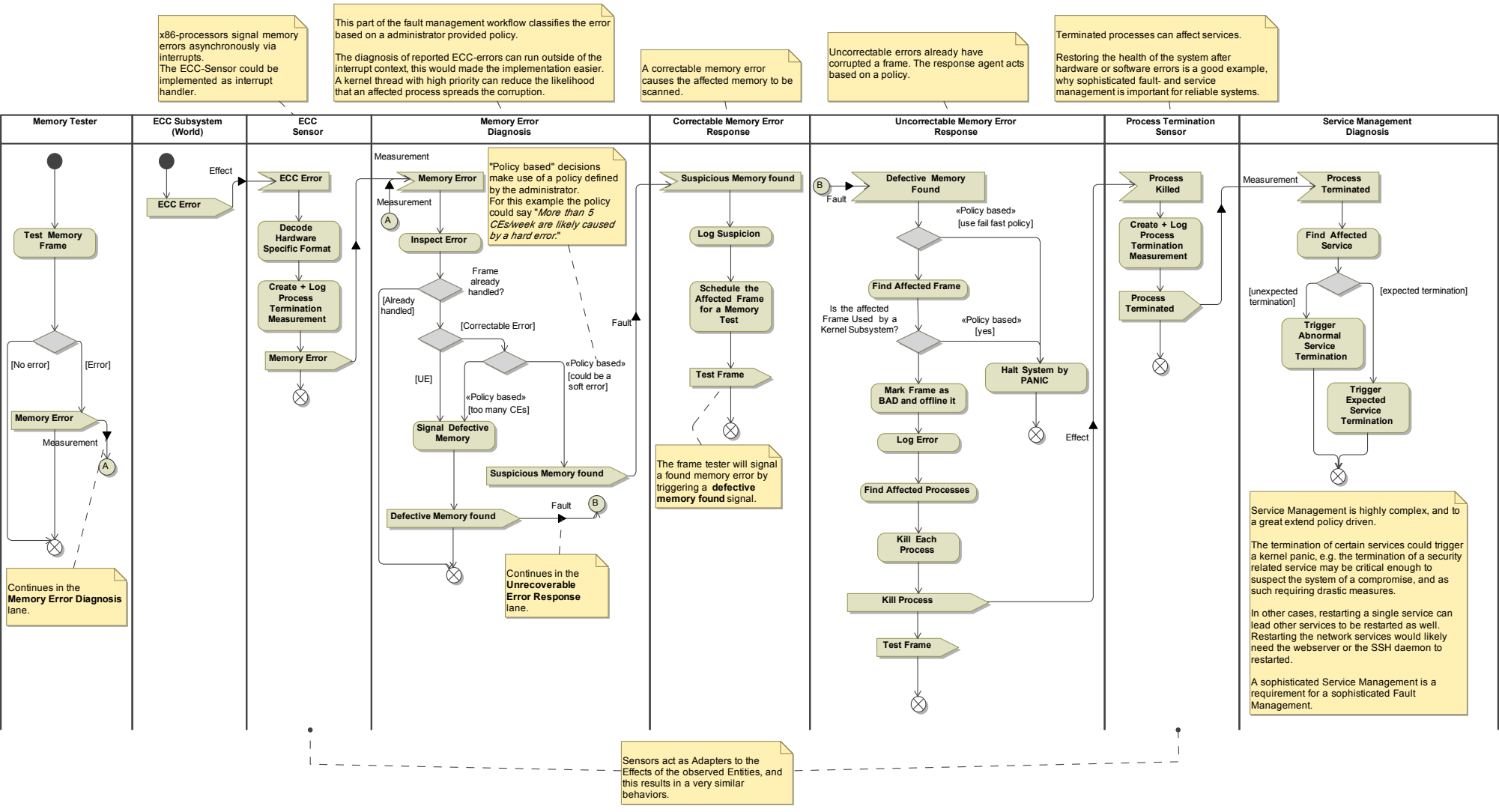
3 a) The "undesired effect"

3 b) The "undesired effect" happens, when one or both of the faults described by the child nodes occur.

3 c) Data corruption by operating system issues or hardware other than the memory subsystem are not further elaborated in this fault tree.



3 d) Note, that this fault tree does not include systems without ECC-correction. Further, this tree does not distinguish between hard- and soft errors.


x86-processors signal memory errors asynchronously via interrupts.
The ECC-Sensor could be implemented as interrupt handler.
This part of the fault management workflow classifies the error based on a administrator provided policy.
The diagnosis of reported ECC-errors can run outside of the interrupt context, this would made the implementation easier.
A kernel thread with high priority can reduce the likelihood that an affected process spreads the corruption.
A correctable memory error causes the affected memory to be scanned.
Uncorrectable errors already have corrupted a frame. The response agent acts based on a policy.
Terminated processes can affect services.
Restoring the health of the system after hardware or software errors is a good example, why sophisticated fault- and service management is important for reliable systems.
Memory Tester
ECC Subsystem (World)
ECC Sensor
Memory Error Diagnosis
Correctable Memory Error Response
Uncorrectable Memory Error Response
Process Termination Sensor
Service Management Diagnosis
Test Memory Frame
[No error]
[Error]
Memory Error
Measurement
A
Continues in the Memory Error Diagnosis lane.
Effect
ECC Error
ECC Error
Decode Hardware Specific Format
Create + Log Process Termination Measurement
Memory Error
Measurement
Memory Error
Measurement
A
Inspect Error
"Policy based" decisions make use of a policy defined by the administrator.
For this example the policy could say "More than 5 CEs/week are likely caused by a hard error."
Frame already handled?
[Already handled]
[Correctable Error]
[UE]
«Policy based» [too many CEs]
«Policy based» [could be a soft error]
Signal Defective Memory
Suspicious Memory found
Fault
Defective Memory found
Fault
B
Continues in the Unrecoverable Error Response lane.
Suspicious Memory found
Log Suspicion
Schedule the Affected Frame for a Memory Test
Test Frame
The frame tester will signal a found memory error by triggering a defective memory found signal.
B
Fault
Defective Memory Found
«Policy based» [use fail fast policy]
Find Affected Frame
Is the affected Frame Used by a Kernel Subsystem?
«Policy based» [yes]
Halt System by PANIC
Mark Frame as BAD and offline it
Log Error
Find Affected Processes
Kill Each Process
Kill Process
Test Frame
Effect
Process Killed
Create + Log Process Termination Measurement
Process Terminated
Measurement
Process Terminated
Find Affected Service
[unexpected termination]
[expected termination]
Trigger Abnormal Service Termination
Trigger Expected Service Termination
Service Management is highly complex, and to a great extend policy driven.
The termination of certain services could trigger a kernel panic, e.g. the termination of a security related service may be critical enough to suspect the system of a compromise, and as such requiring drastic measures.
In other cases, restarting a single service can lead other services to be restarted as well. Restarting the network services would likely need the webserver or the SSH daemon to restarted.
A sophisticated Service Management is a requirement for a sophisticated Fault Management.
Sensors act as Adapters to the Effects of the observed Entities, and this results in a very similar behaviors.

Related Work: Speicher(tests)

technische universität dortmund

Related Work: Speicher(tests)

Related Work: Software

technische universität dortmund

Related Work: Software

- Memtest86+
- Online Test für Solaris 8[3]
- EffoGPL
- Linux EDAC / Linux HW-Poison / mcelog
- Solaris Fault Management

[3] Singh et al. *Software Based In-System Memory Test For Highly Available Systems* , 2005
DOI: 10.1109/IOLTS.2005.13

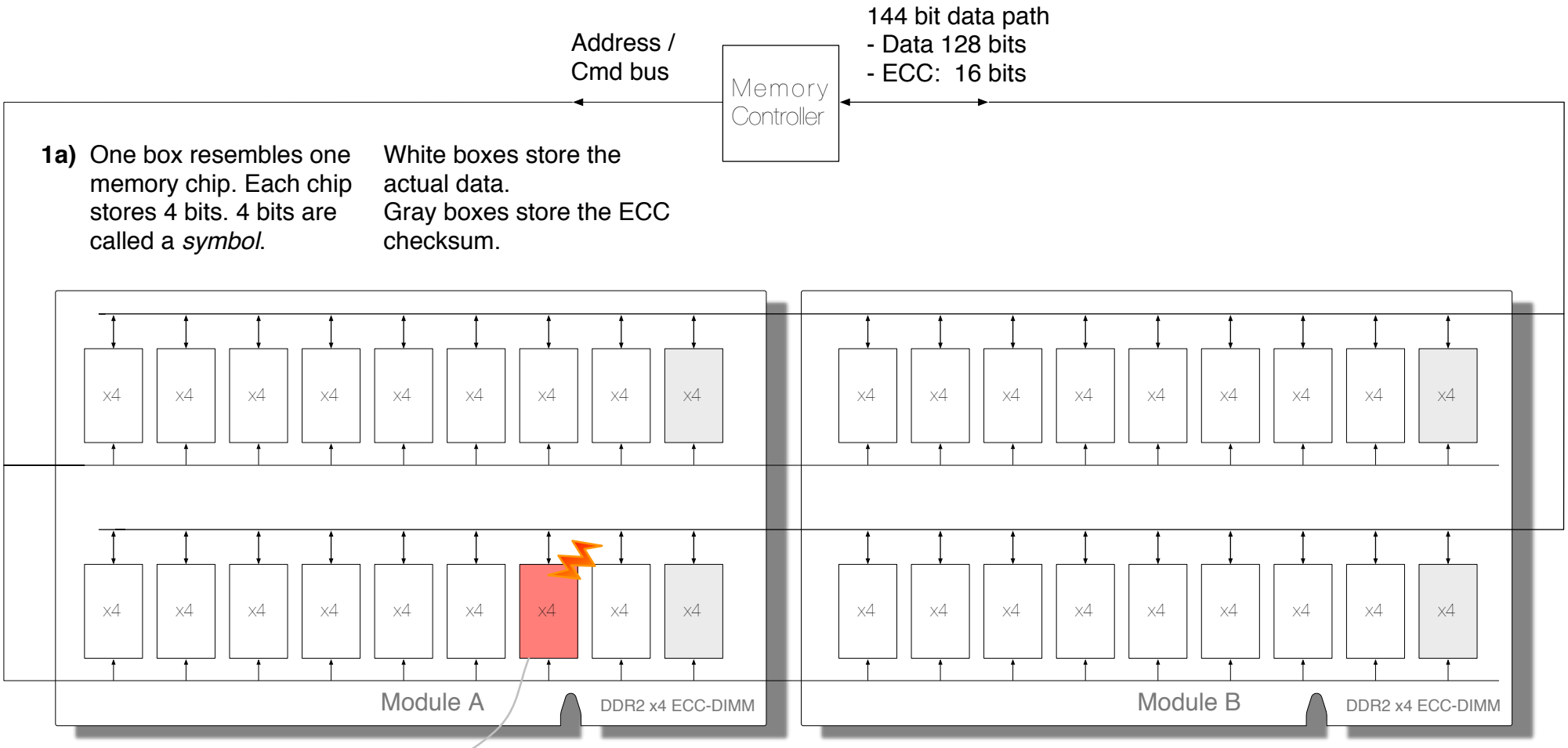
Related Work: Hardware

technische universität dortmund

Related Work: Hardware

- ECC
- ChipKill - Ansatz
- Virtualized ECC[4]

[4] Yoon, Doe H. ; Erez, Mattan: Virtualized and flexible ECC for main memory. In: SIGARCH Comput. Archit. News 38 (2010), March, 397–408



1a) One box resembles one memory chip. Each chip stores 4 bits. 4 bits are called a *symbol*.

White boxes store the actual data.
Gray boxes store the ECC checksum.

2a) A defective memory chip results in the loss of a 4 bit symbol.

2b) Four 4-bit check symbols provide SSC-DSD protection for 32 4-bit data symbols, allowing the defect chip to be compensated.

Proactive Memory Tests . . .

technische universität dortmund

Proactive Memory Tests for the Linux Kernel

Idee

technische universität dortmund

Die Idee

- Online Speichertests sind wichtig
- Grade für Systeme ohne/mit unzureichender HW-Unterstützung
- Kernelentwicklung hat viele Nachteile
  - Sehr Kernel-Version abhängig
  - Debugging
  - Festlegung auf *C*
- Eine Hybridlösung User/Kernel würde das Beste beider Welten vereinen können

Proactive Memory Tests . . .

technische universität dortmund

Proactive Memory Tests for the Linux Kernel

Anforderungen

Detection of Errors

technische universität dortmund

Detection of Errors

- The implementation must be able to detect memory errors.
- . . . should be able to test a substantial amount of the physical memory.
- . . . does not need to be able to test the physical memory that is used by the kernel for its internal data and code.

Detection of Errors

technische universität dortmund

Detection of Errors II

- . . . should be able be able to test memory, that is marked as free by the Buddy-Allocator.
- . . . should be able be able to test memory that is used as anonymous memory. It is not necessary, that memory that is backed by swap space or memory mapped files can be tested.
- . . . should be able be able to test memory, that is used in the Linux page cache. Memory that is marked as *dirty*, or memory that is currently used by I/O does not need to be tested.

Different Fault Models

technische universität dortmund

Different Fault Models

- The implementation must support multiple fault models. It is sufficient, if only a restricted subset of all possible fault models can be supported by the implementation.
- . . . must implement an emulation that resembles the runtime complexity of two fault models.
- . . . must support fault models that test for NPSFs. The implementation is allowed to restrict size and composition of the neighbourhoods.

Cost-Benefit Ratio

technische universität dortmund

Cost-Benefit Ratio

- By benchmarking the application it should be possible to estimate the cost of an online memory test.
- The implementation should provide performance metrics that indicate how expensive the acquisition of certain memory areas has been.
- By benchmarking the application it should be possible to estimate, how big the impact of memory testing is.

Base for Further Research

technische universität dortmund

Base for Further Research

- The implementation should be written in such a way, that further further research can adopt and change it easily.
- . . . should provide the means to visualize the state of memory over time.

Performance

technische universität dortmund

Performance

- The implementation should be written in such a way, that it is feasible to detect performance bottlenecks.
- . . . should be written in such a way, that it is feasible to apply high level performance optimizations.

Extensibility

technische universität dortmund

Extensibility

- The implementation should be written in such a way, that it easily extensible.
- . . . should be designed in a modular fashion.

Stability

Stability

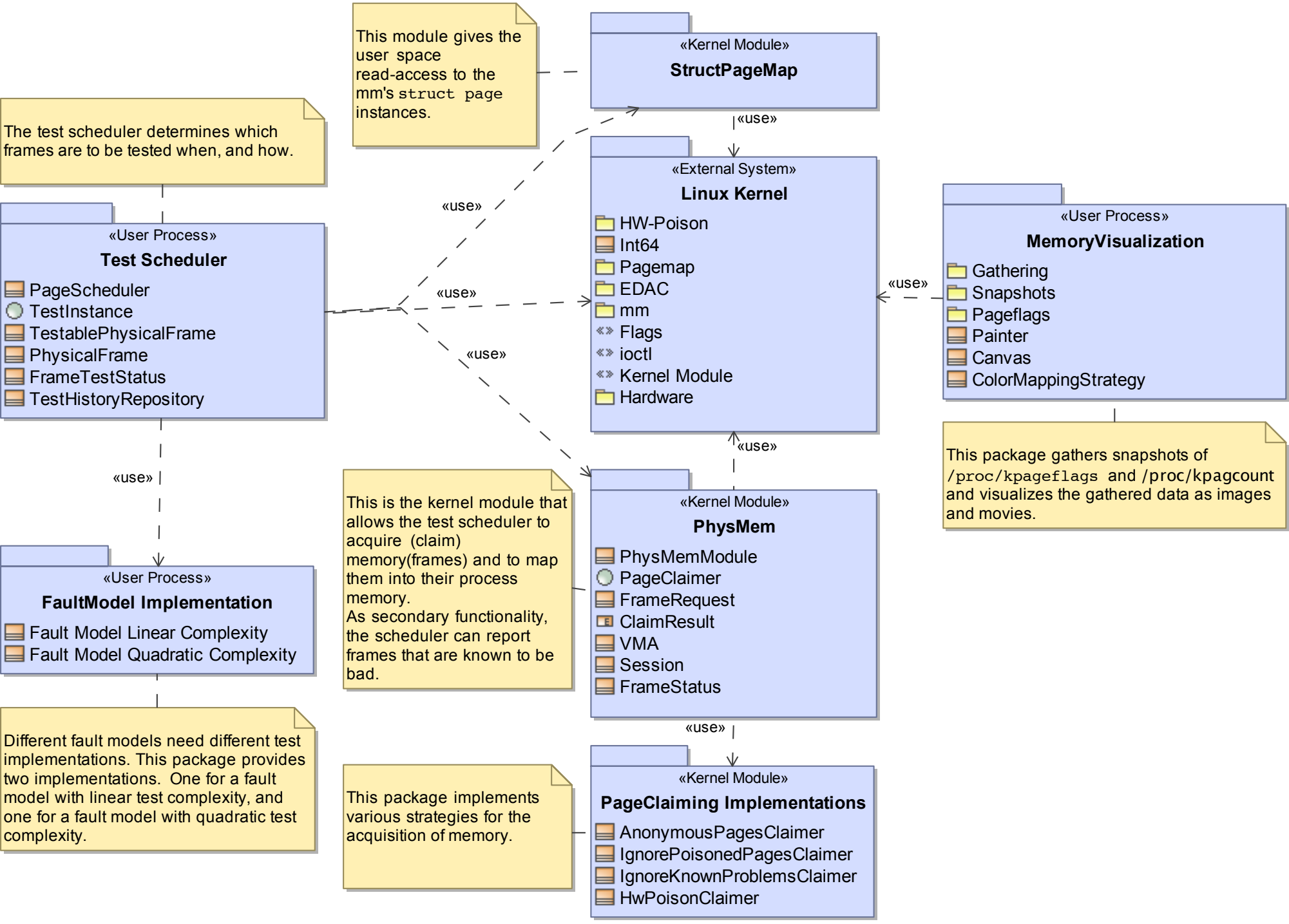
- The components of the system should be isolated from each other, so that faults remain local to single components.
- The design must ensure that memory leaks are effectively prevented.
- No component should have more privileges, than it absolutely needs to function.
- It should be easy to test the implementation.

Maintainability

technische universität dortmund

Maintainability

- The implementation should be easily maintainable, and be impervious to changes in the kernel.
- . . . should not break the encapsulation of the kernel's internal data structures, unless absolutely neccessary.

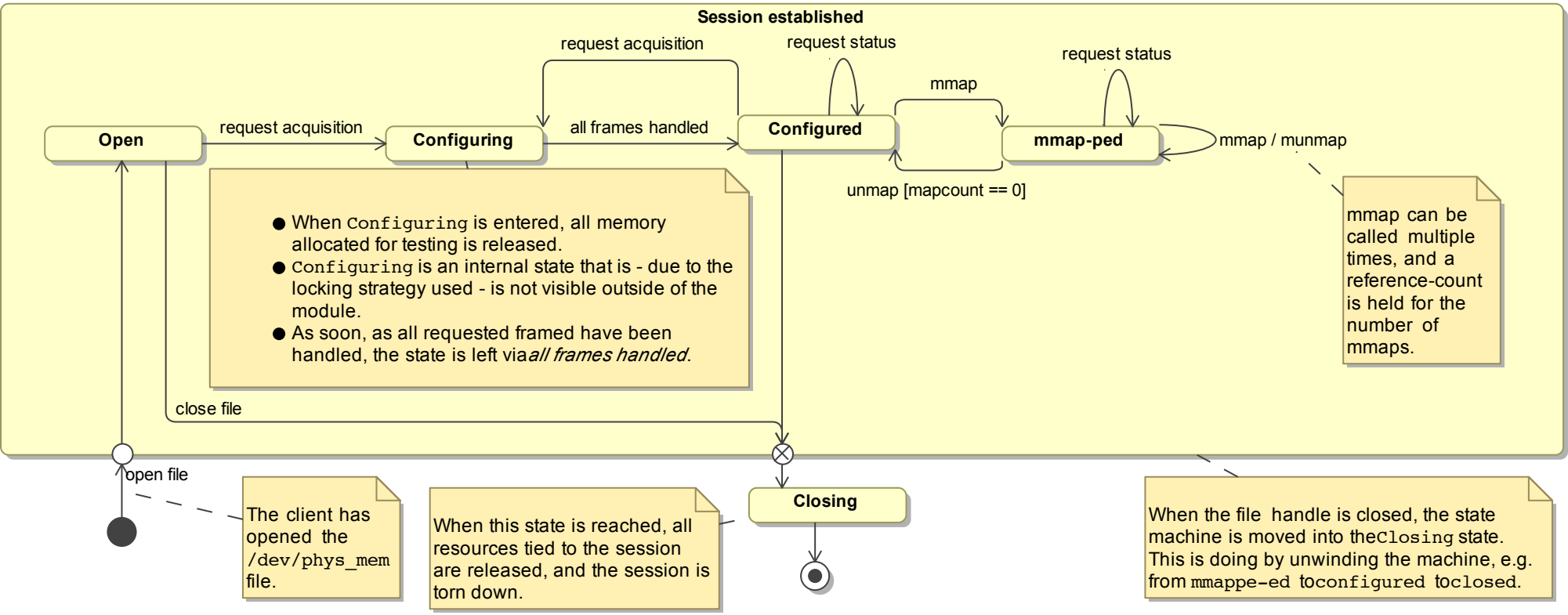

This module gives the user space read-access to the mm's struct page instances.
«Kernel Module»
StructPageMap
«use»
The test scheduler determines which frames are to be tested when, and how.
«External System»
Linux Kernel
HW-Poison
Int64
Pagemap
EDAC
mm
Flags
ioctl
Kernel Module
Hardware
«use»
«User Process»
MemoryVisualization
Gathering
Snapshots
Pageflags
Painter
Canvas
ColorMappingStrategy
«use»
«User Process»
Test Scheduler
PageScheduler
TestInstance
TestablePhysicalFrame
PhysicalFrame
FrameTestStatus
TestHistoryRepository
«use»
«use»
«use»
«use»
This package gathers snapshots of /proc/kpageflags and /proc/kpagcount and visualizes the gathered data as images and movies.
This is the kernel module that allows the test scheduler to acquire (claim) memory(frames) and to map them into their process memory.
As secondary functionality, the scheduler can report frames that are known to be bad.
«Kernel Module»
PhysMem
PhysMemModule
PageClaimer
FrameRequest
ClaimResult
VMA
Session
FrameStatus
«use»
«User Process»
FaultModel Implementation
Fault Model Linear Complexity
Fault Model Quadratic Complexity
Different fault models need different test implementations. This package provides two implementations. One for a fault model with linear test complexity, and one for a fault model with quadratic test complexity.
This package implements various strategies for the acquisition of memory.
«Kernel Module»
PageClaiming Implementations
AnonymousPagesClaimer
IgnorePoisonedPagesClaimer
IgnoreKnownProblemsClaimer
HwPoisonClaimer

Implementierung

technische universität dortmund

Kernel

- Speicher über Strategy-Implementierungen "‘allozieren"’
- Konfiguration via *IOCTL*
- Speicher wird pro *Session* verwaltet
- Thread-Safe implementiert

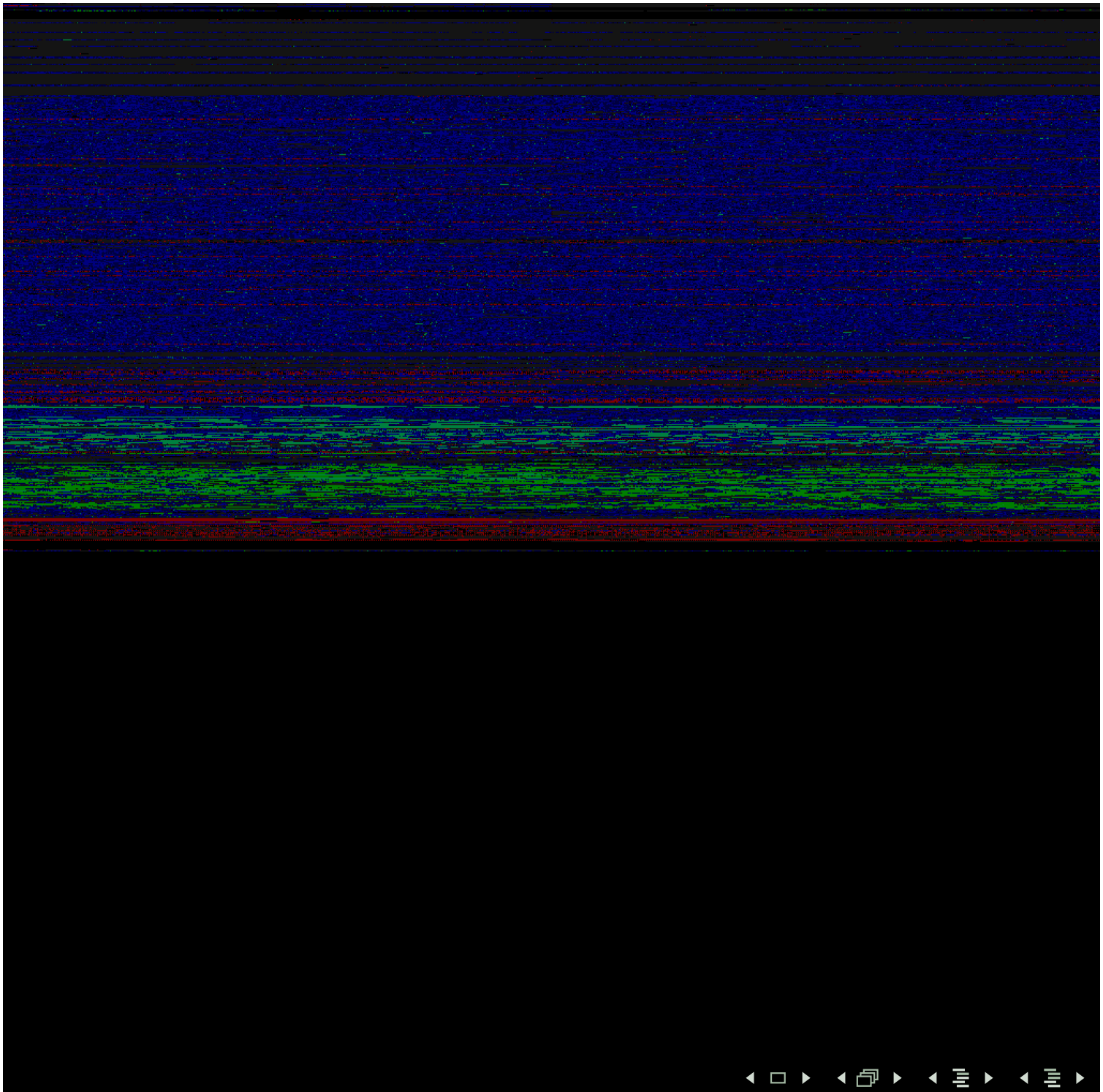

Session established
request acquisition
request status
request status
mmap
Open
request acquisition
Configuring
all frames handled
Configured
mmap-ped
mmap / munmap
unmap [mapcount == 0]
● When Configuring is entered, all memory allocated for testing is released.
● Configuring is an internal state that is - due to the locking strategy used - is not visible outside of the module.
● As soon, as all requested framed have been handled, the state is left via all frames handled.
mmap can be called multiple times, and a reference-count is held for the number of mmaps.
close file
open file
The client has opened the /dev/phys_mem file.
When this state is reached, all resources tied to the session are released, and the session is torn down.
Closing
When the file handle is closed, the state machine is moved into the Closing state. This is doing by unwinding the machine, e.g. from mmappe-ed to configured to closed.

Implementierung

technische universität dortmund

Userland

- Implementiert in Python
- Schnittstelle zum Kernel Modul in Library
- Diverse Analyse Tools
- Testergebnisse werden persistiert



Proactive Memory Tests . . .

technische universität dortmund

Proactive Memory Tests for the Linux Kernel
. . . Herausforderungen

Probleme

technische universität dortmund

Userland

- Schnittstelle zum Kernel Modul in Library "‘unschön"’

Probleme

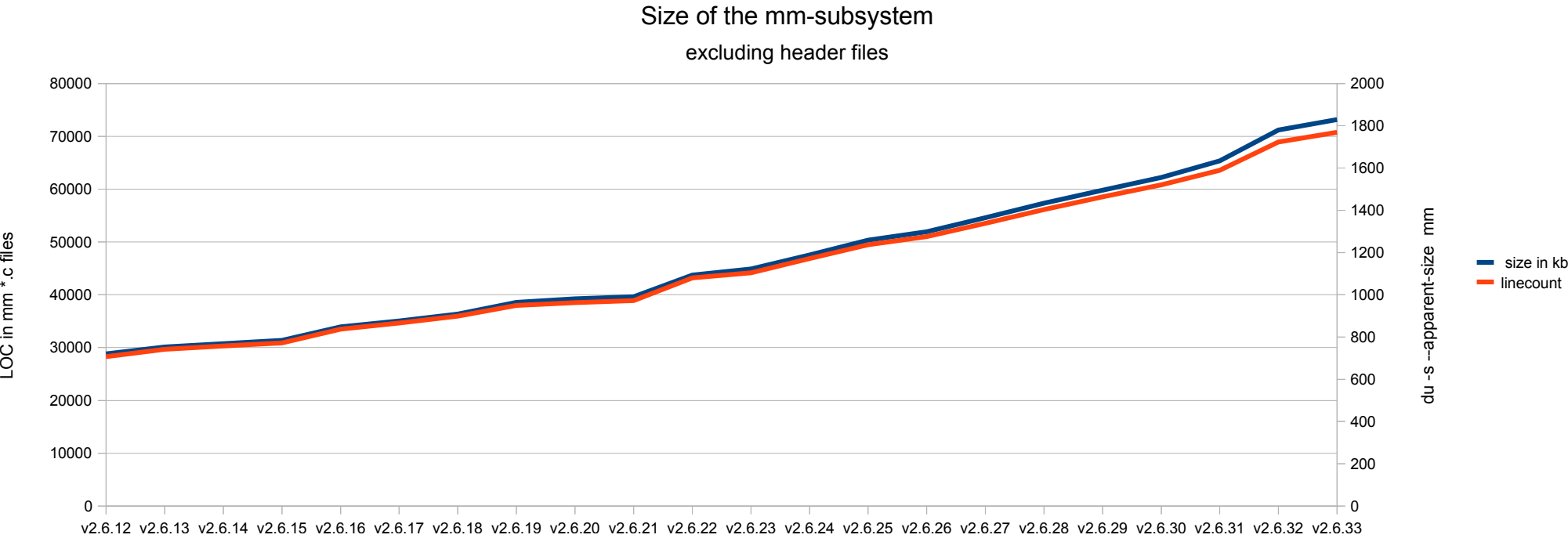
technische universität dortmund

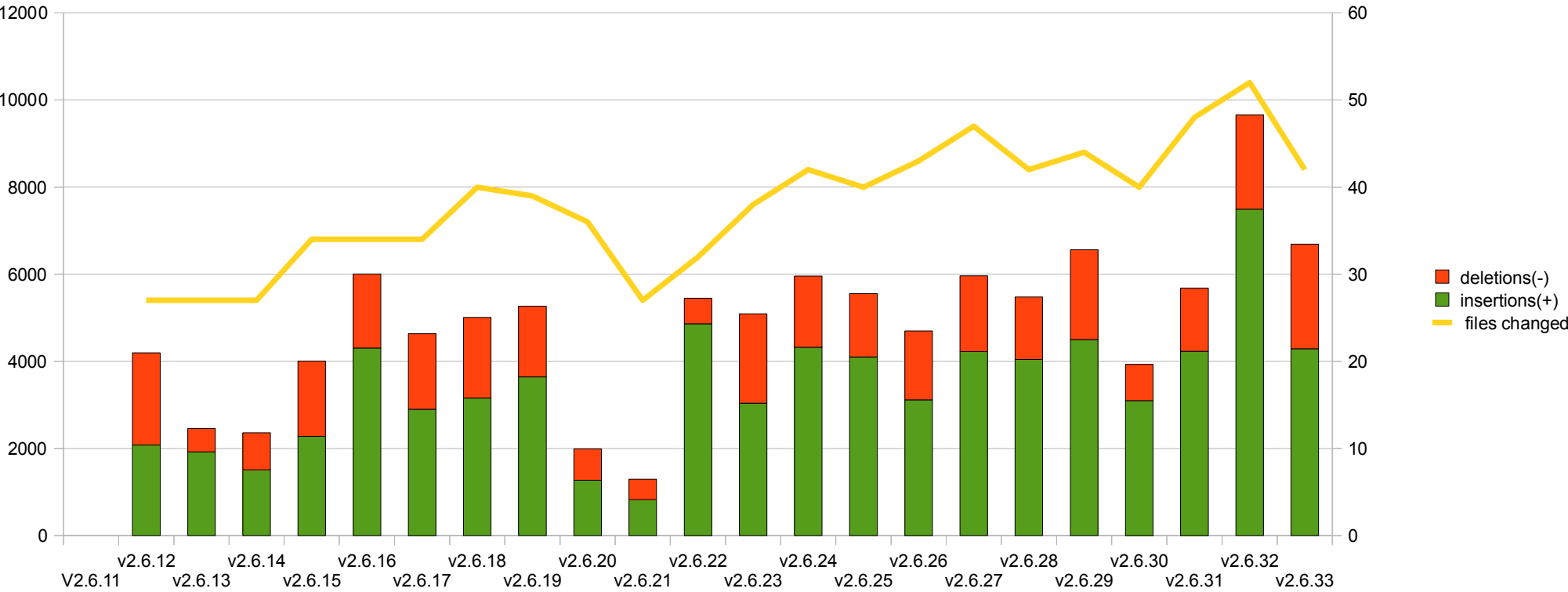
Kernel

- Bücher oft zu alt
- MM *sehr* volatil
- MM *sehr* schlecht dokumentiert
- MM *sehr* groß – *Understanding the Linux Virtual Memory Manager* hat **748** Seiten
- Selbst die, die sich auskennen sollten machen Fehler (mcelog + HW-Poison immer noch nicht stabil)

Testen

- FAUmachine **sehr** langsam
- KVM und FAUmachine **sollten** das selbe Image Format nutzen. . .


Size of the mm-subsystem
excluding header files
LOC in mm *.c files
du -s --apparent-size mm
size in kb
linecount
80000
70000
60000
50000
40000
30000
20000
10000
0
2000
1800
1600
1400
1200
1000
800
600
400
200
0
v2.6.12 v2.6.13 v2.6.14 v2.6.15 v2.6.16 v2.6.17 v2.6.18 v2.6.19 v2.6.20 v2.6.21 v2.6.22 v2.6.23 v2.6.24 v2.6.25 v2.6.26 v2.6.27 v2.6.28 v2.6.29 v2.6.30 v2.6.31 v2.6.32 v2.6.33


12000
10000
8000
6000
4000
2000
0
60
50
40
30
20
10
0
V2.6.11
v2.6.12
v2.6.13
v2.6.14
v2.6.15
v2.6.16
v2.6.17
v2.6.18
v2.6.19
v2.6.20
v2.6.21
v2.6.22
v2.6.23
v2.6.24
v2.6.25
v2.6.26
v2.6.27
v2.6.28
v2.6.29
v2.6.30
v2.6.31
v2.6.32
v2.6.33
deletions(-)
insertions(+)
files changed

Ergebnisse

technische universität dortmund

Ergebnisse

Ergebnisse

technische universität dortmund

Ergebnisse

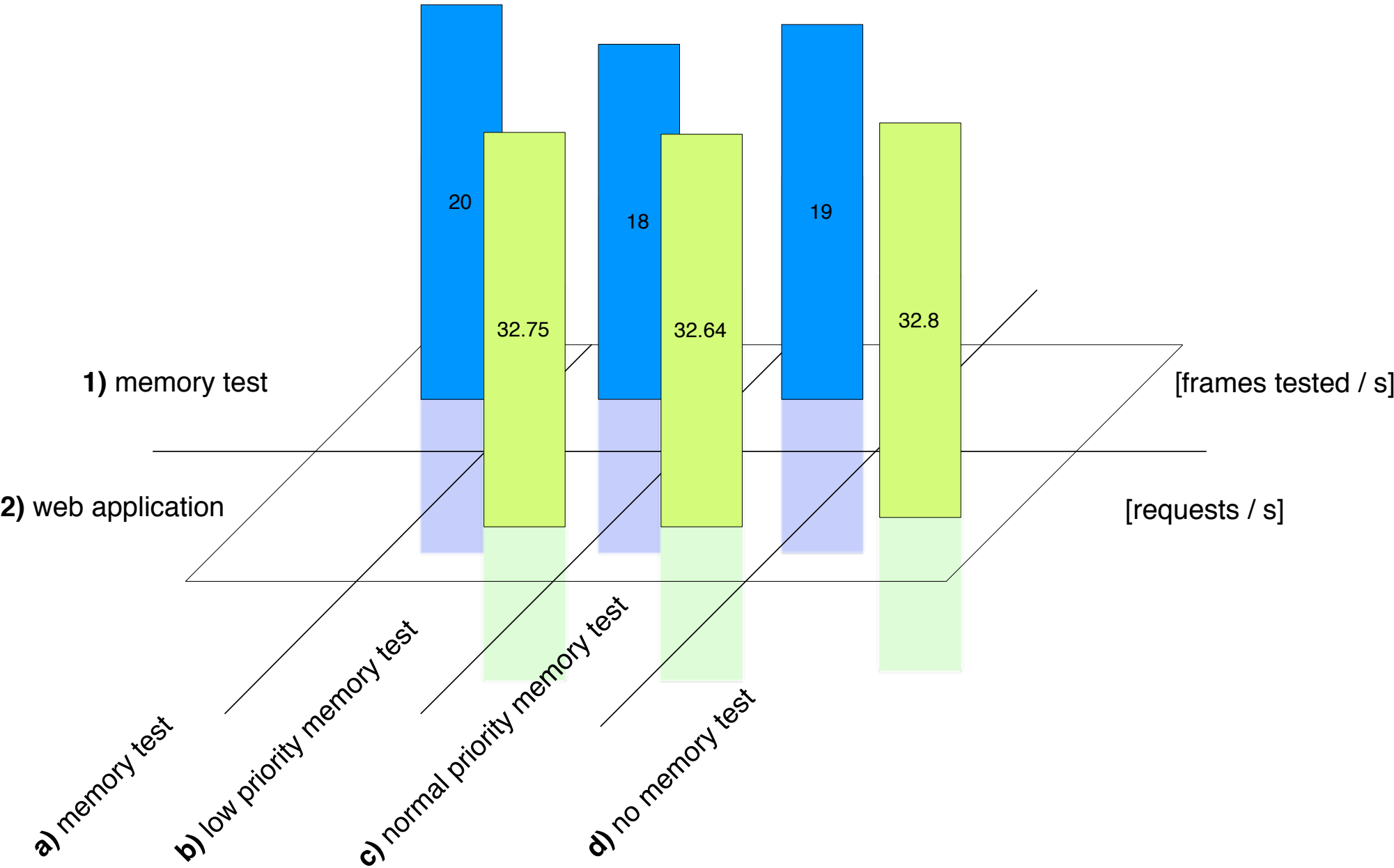
- Nahezu alle Ziele erreicht
  - Anon-User und Page-Cache konnte nicht zuverlässig behandelt werden.
  - Buddy-Allocator funktioniert stabil!
- Saubere Architektur, sauber implementiert
  - Unit-Tests fuer Kernel und Userland
- Test mit FAUmachine erfolgreich
- Abdeckung: $5,44$ von 8 *GiB*

Ergebnisse im Detail

technische universität dortmund

Test: Web Application (Drupal)

- Test: Web Application (Drupal)
- Last mit *ab* erzeugen
- Tests wurden jeweils 5 mal wiederholt, das Ergebnis ist der Median der Messergebnisse
  - Wird das Ergebnis aus dem Cache geliefert, dann wird die Performance kaum beeinträchtigt
- Last mit *skipfish* erzeugen
  - Hohe CPU-Last wird erzeugt
  - *nice* für Memory-Test erlaubt trotzdem hohen Durchsatz der Web-App.
- Fazit: Der Speichertest lässt sich sehr schön durch BS Mittel steuern.

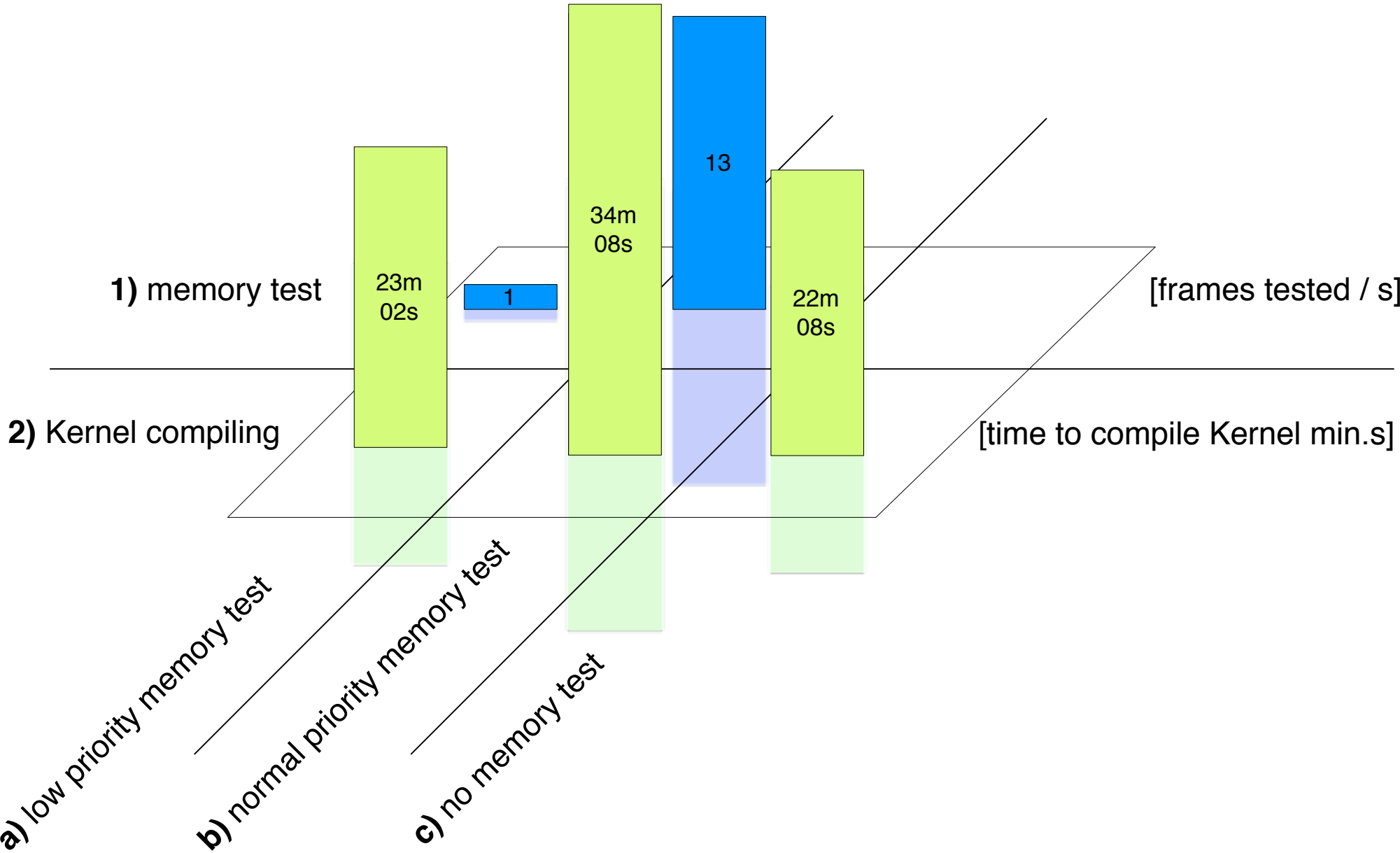

20
32.75
18
32.64
19
32.8
1) memory test
[frames tested / s]
2) web application
[requests / s]
a) memory test
b) low priority memory test
c) normal priority memory test
d) no memory test

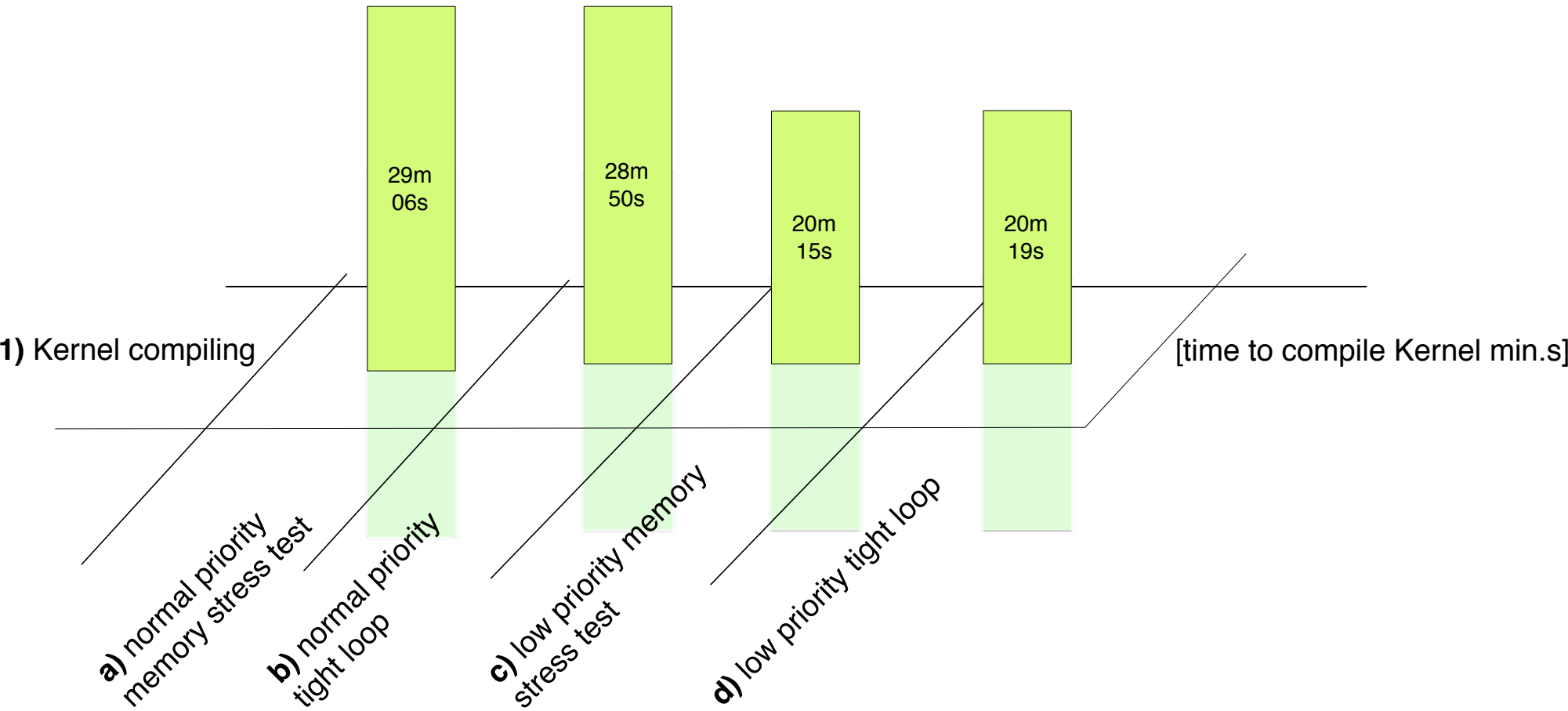
Ergebnisse im Detail

technische universität dortmund

Test: Kernel compilieren

- Test: Kernel compilieren ($-j2$)
- Tests wurden jeweils 5 mal wiederholt, das Ergebnis ist der Median der Messergebnisse
  - Einfluss des Tests istr deutlich spürbar.
  - Hohe CPU-Last wird erzeugt
  - *nice* für Memory-Test erlaubt trotzdem hohen Geschwindigkeit.
- Fazit: Sehr CPU und speicherintensive Anwendungen konkurrieren mit dem Speichertest.
- Das wird auch durch einen Mikro-Benchmark unterstützt.


1) memory test
[frames tested / s]
23m
02s
1
34m
08s
13
22m
08s
2) Kernel compiling
[time to compile Kernel min.s]
a) low priority memory test
b) normal priority memory test
c) no memory test


29m
06s
28m
50s
20m
15s
20m
19s
1) Kernel compiling
[time to compile Kernel min.s]
a) normal priority
memory stress test
b) normal priority
tight loop
c) low priority memory
stress test
d) low priority tight loop

Aus- und Rückblick

technische universität dortmund

Aus- und Rückblick

technische universität dortmund

Und jetzt? – Ausblick

- Speichertests sind wichtig.
- In der Linux Bewegung kommt diese Erkenntnis mittlerweile an.
- Speichertests lassen sich wahrscheinlich gut in einen Hypervisor integrieren.
- Selbiges gilt für Microkernel-Architekturen
  - Dort sind sie auch wichtig.
- Hybrid-Ansätze (*Virtualized ECC*) sehen vielversprechend aus.

technische universität dortmund

Und jetzt? – Rückblick

- Die Linux mm könnte eine Umstrukturierung vertragen
  - Viele gute Features: Page Migration, Page Deduplication, Hohe Performance
  - Schlechte Dokumentation.
  - Keine Trennung der Aspekte
  - Viele implizite Abhängigkeiten
- Vielen Dank für das Vertrauen und die Hilfe!
- Was kam heute nicht vor?
  - Interna der mm, Diskussion: NUMA/UMA, Multi-Kern/Multi-Prozessor, Solaris, Quelltexte, Detailierte Fehlermodelle, . . .

technische universität dortmund

Fragen? Fragen!

From the NASA website, updated May 17, 2010 at 5:00 PT:

> *One flip of a bit in the memory of an onboard computer appears to have caused the change in the science data pattern returning from Voyager 2, engineers at NASA's Jet Propulsion Laboratory said Monday, May 17. A value in a single memory location was changed from a 0 to a 1.*