TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Systems Architecture, Operating Systems Group

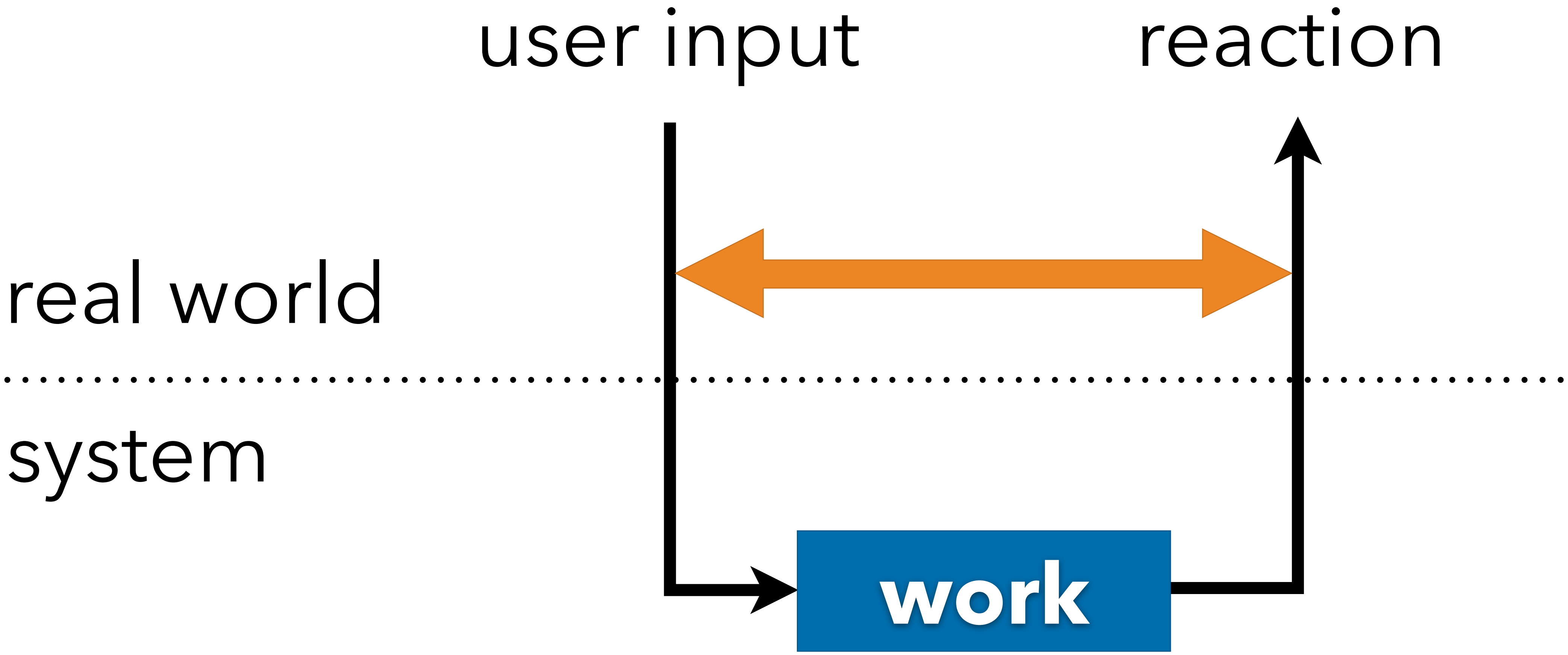
REAL-TIME SYSTEMS

MICHAEL ROITZSCH

DEFINITION

system whose quality depends on the **functional correctness** of computations and the **time** those results are produced

RELEVANCE



LINES OF THOUGHT

real-time school	**systems school**
mathematically sound	practical systems
simplifying assumptions	baroque details
random task sets	actual applications
strong contract	usable interface
theory	implementation

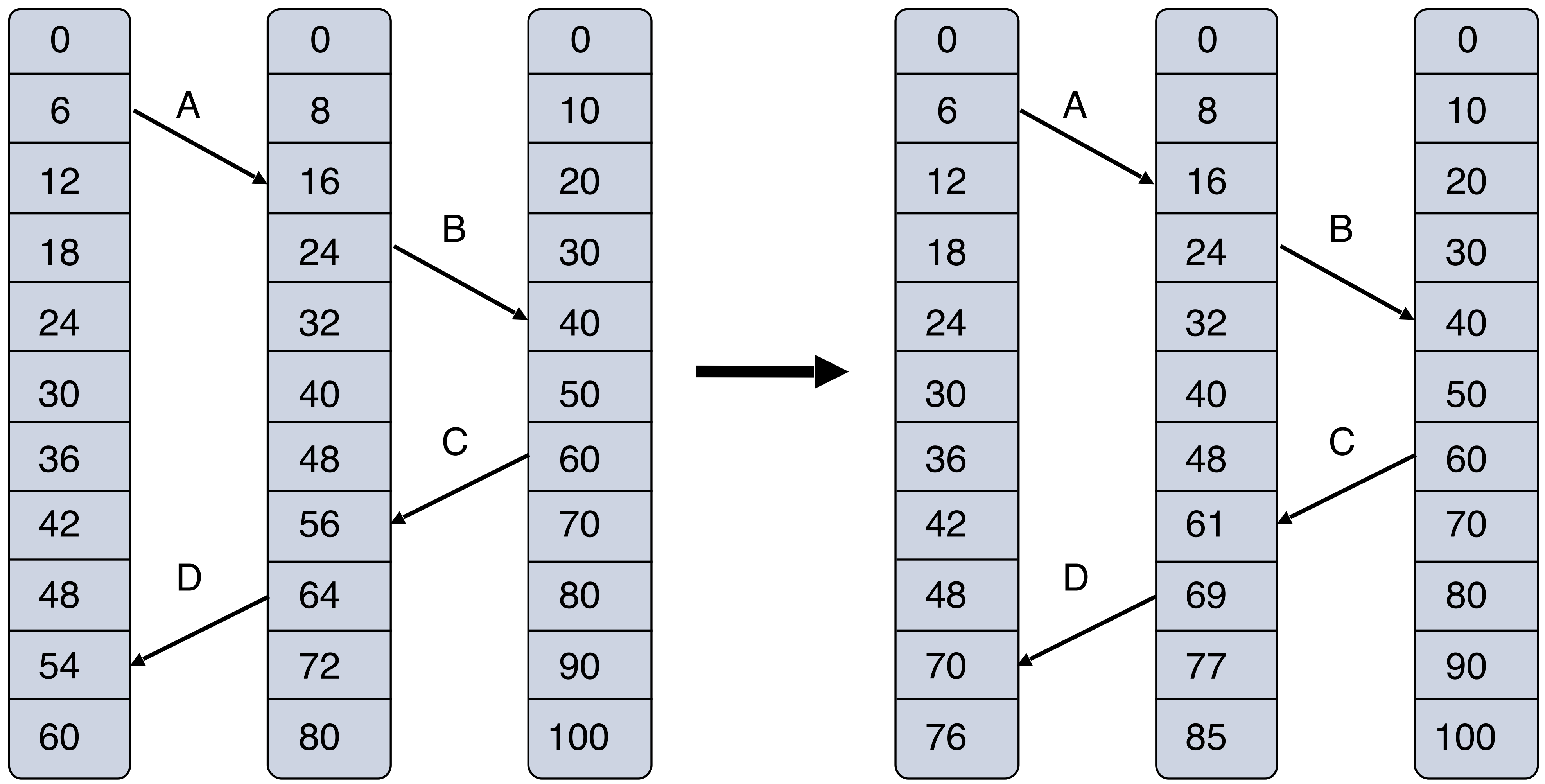
THE OLD DAYS

Leslie Lamport: Time, Clocks, and the Ordering of Events in a Distributed System. Communications of the ACM 21(7): pp. 558–565, July 1978

HoF

Perhaps the first true “distributed systems” paper, it introduced the concept of “causal ordering”, which turned out to be useful in many settings. The paper proposed the mechanism it called “logical clocks”, but everyone now calls these “Lamport clocks.”

LOGICAL CLOCKS



0	0	0
6	8	10
12	16	20
18	24	30
24	32	40
30	40	50
36	48	60
42	56	70
48	64	80
54	72	90
60	80	100

0	0	0
6	8	10
12	16	20
18	24	30
24	32	40
30	40	50
36	48	60
42	61	70
48	69	80
70	77	90
76	85	100

THEORY

Chang L. Liu, James W. Layland: Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. Journal of the ACM, Volume 20(1): pp. 46–61, January 1973

- first description of RMS and EDF
- including the now well-known utilization bounds
- with proof

PRACTICE

Clifford W. Mercer, Stefan Savage, Hideyuki Tokuda: Processor Capacity Reserves: Operating System Support for Multimedia Applications. Proceedings of the IEEE International Conference on Multimedia Computing and Systems (MCS), pp. 90–99, May 1994

- introduced the idea of tracking actual execution time to police overruns
- the reservation concept was born
- implemented in RT-Mach

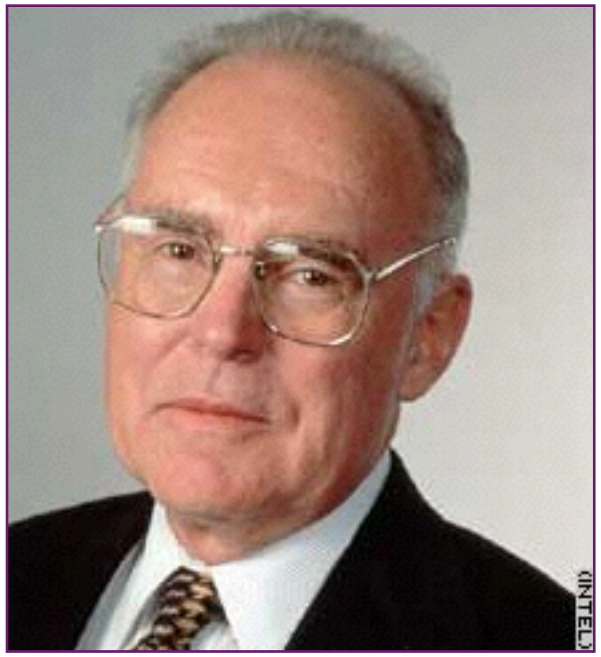
RESOURCE KERNEL

Raj Rajkumar, Kanaka Juvva, et al.: Resource Kernels: A Resource-Centric Approach to Real-Time and Multimedia Systems. Proceedings of the 1998 Multimedia Computing and Networking Conference (MMCN), pp. 150–164, January 1998

- time as a first-class, globally managed shared resource
- based on the periodic task model
- extended with inheritance, reservations

DISASTER STRIKES

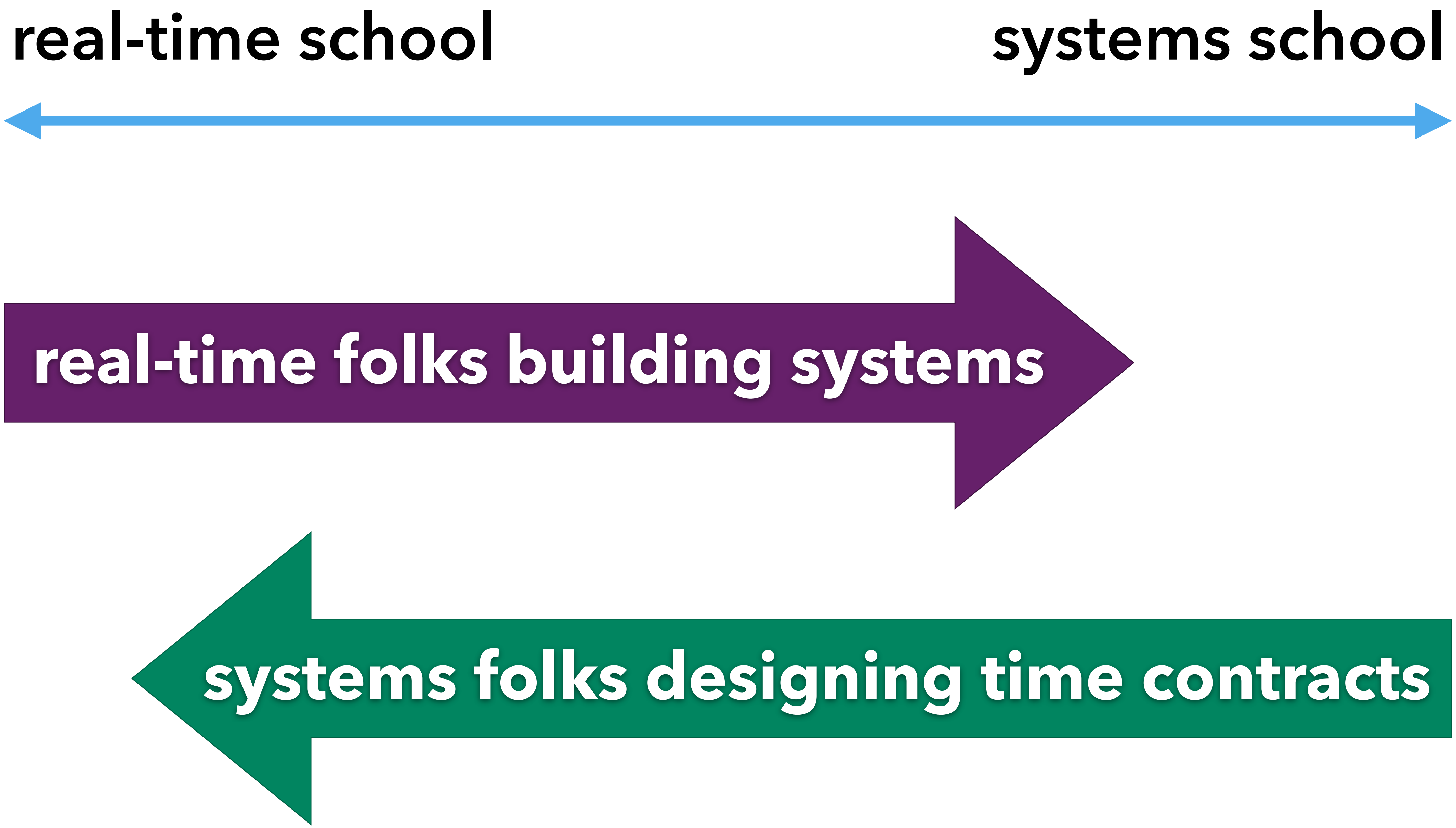
- the 90’s: multimedia driving practical real-time work
- this could have been our world, but then…



(INTEL)

… CPUs got fast.

DIRECTIONS



real-time school

systems school

real-time folks building systems

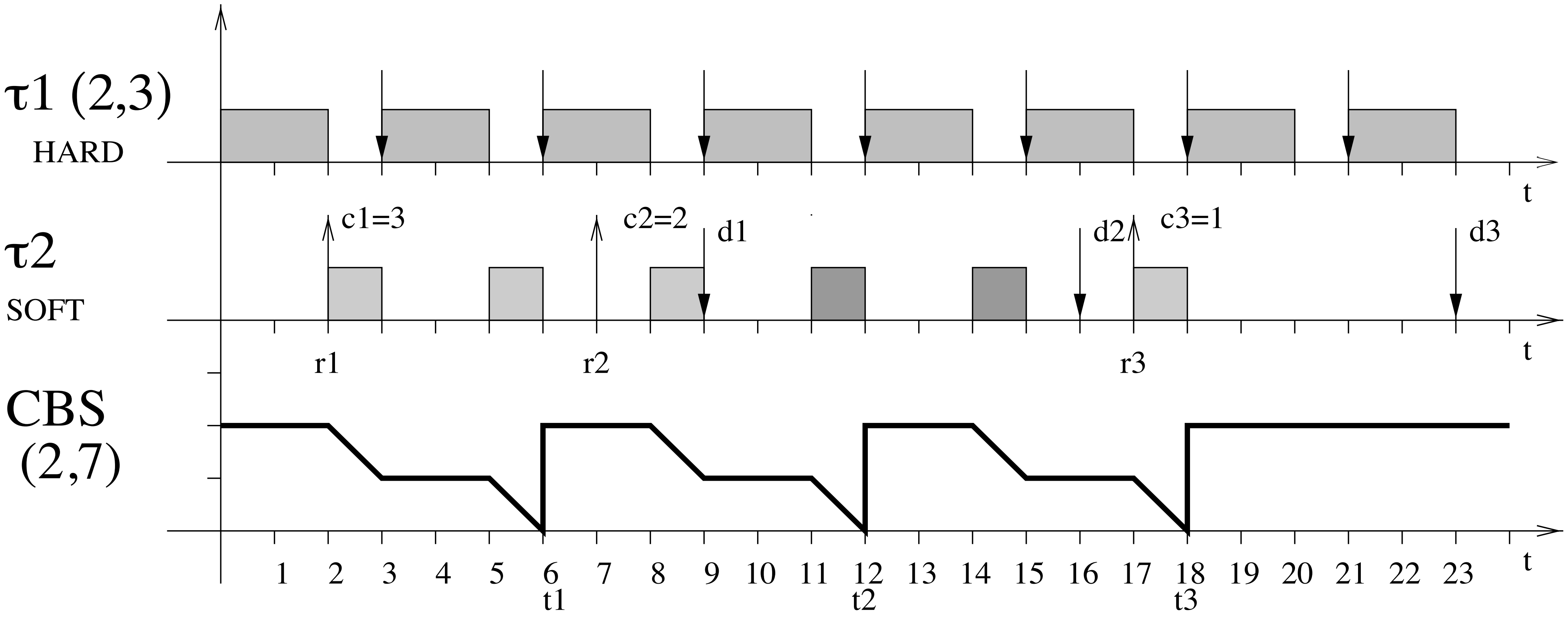
systems folks designing time contracts

CBS

Luca Abeni, Giorgio Buttazzo: Integrating Multimedia Applications in Hard Real-Time Systems. Proceedings of the 19th IEEE Real-Time Systems Symposium (RTSS), pp. 4–13, December 1998

- "server" as a real-time concept
- allocation of time with period and budget
- jobs enqueue to be run by the server
- deadlines postponed on budget overrun

CBS



- illusion of a dedicated, slower processor
- virtual fluid-flow of CPU time
- individual job deadlines hard to manage

SLACK

Luca Marzario, Giuseppe Lipari, et al.: IRIS: A New Reclaiming Algorithm for Server-Based Real-Time Systems. Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 211–218, May 2004

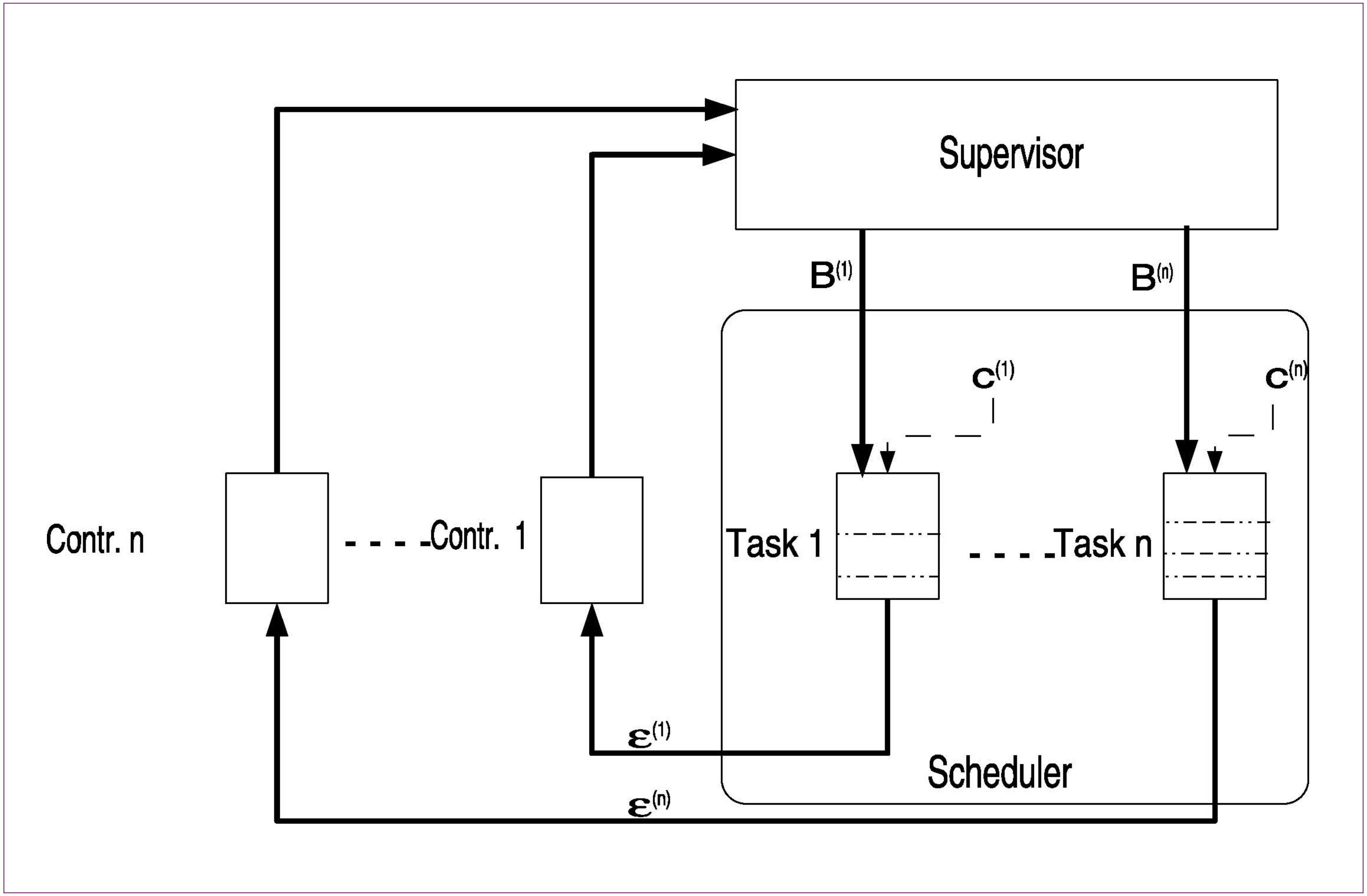
- “slack time” ≈ nothing urgent to do
- redirect slack to help out on overruns

ADAPTING

Luca Abeni, Tommaso Cucinotta, et al.: QoS Management Through Adaptive Reservations. Real-Time Systems, Volume 29(2): pp. 131–155, March 2005

- adapt server budget to varying application demand
- control loop observes tasks,
QoS manager tunes parameters

ADAPTING

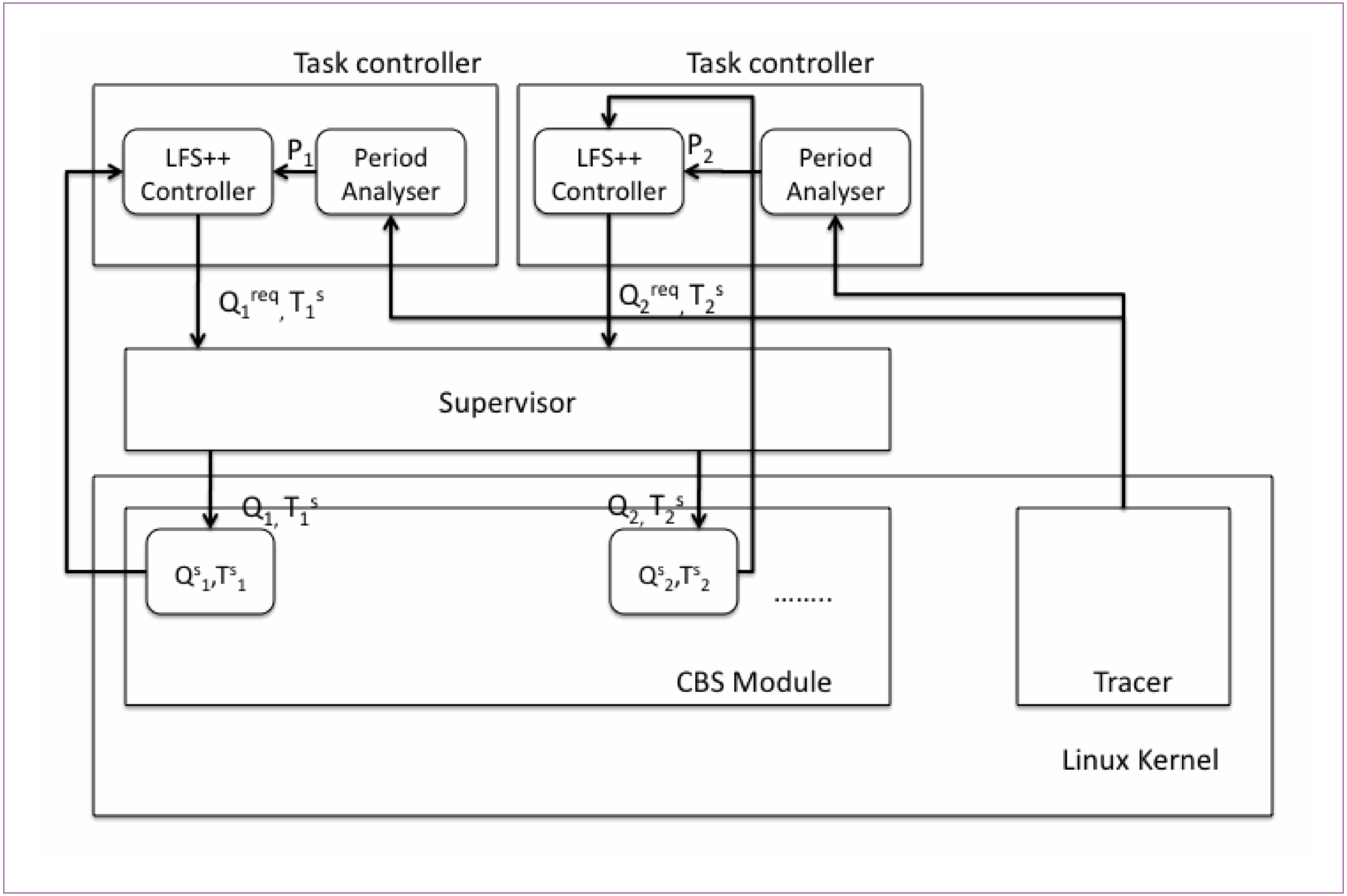

Supervisor
B(1)
B(n)
C(1)
C(n)
Contr. n
- - - -Contr. 1
Task 1
- - - -Task n
ε(1)
ε(n)
Scheduler

GUESSING

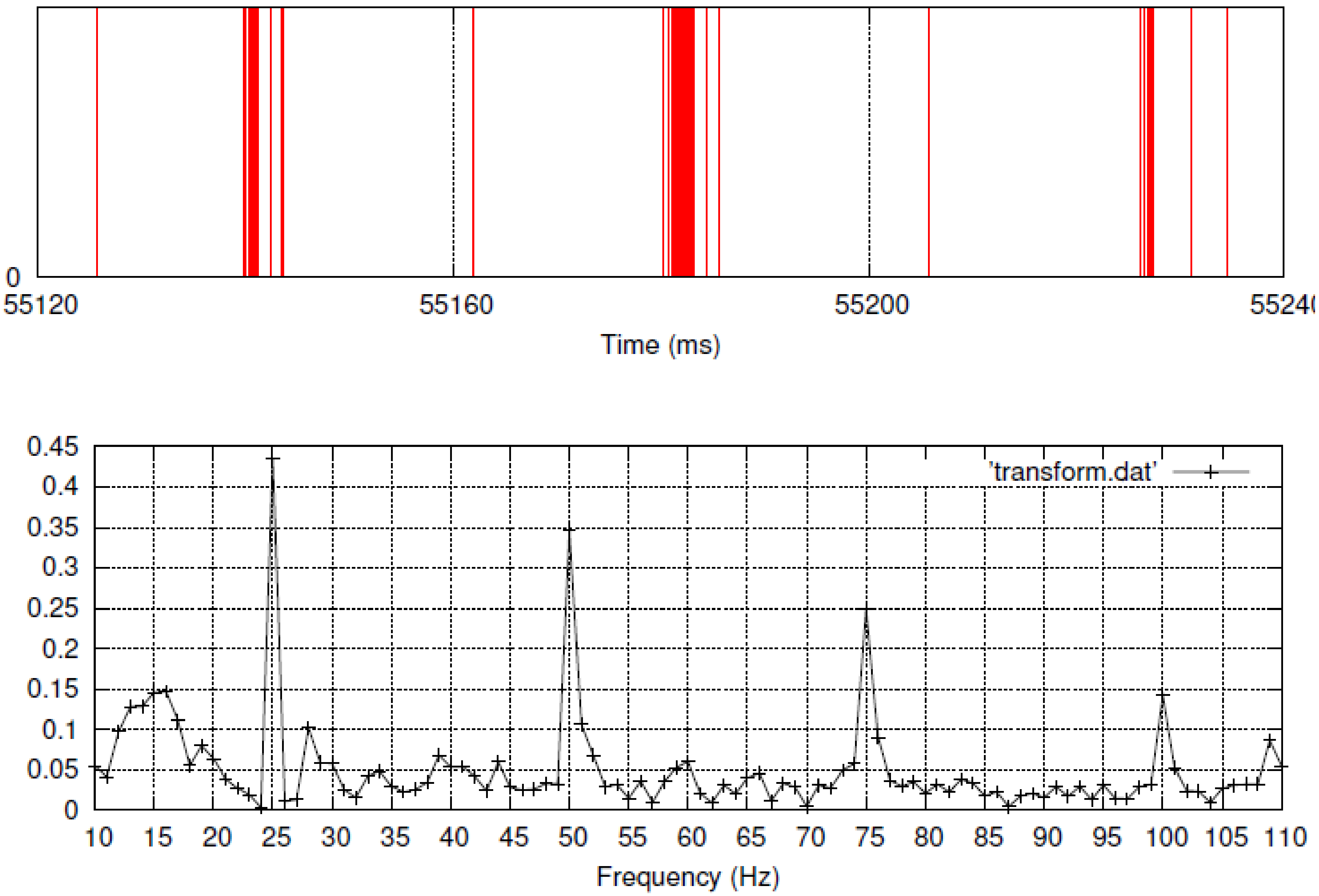
Tommaso Cucinotta, Fabio Checconi, et al.: Self-Tuning Schedulers for Legacy Real-Time Applications. Proceedings of the 5th ACM European Conference on Computer Systems (EuroSys), pp. 55–68, April 2010

- sample system call behavior to infer period and execution time
- provide real-time scheduling to unmodified non-real-time applications

INFRASTRUCTURE

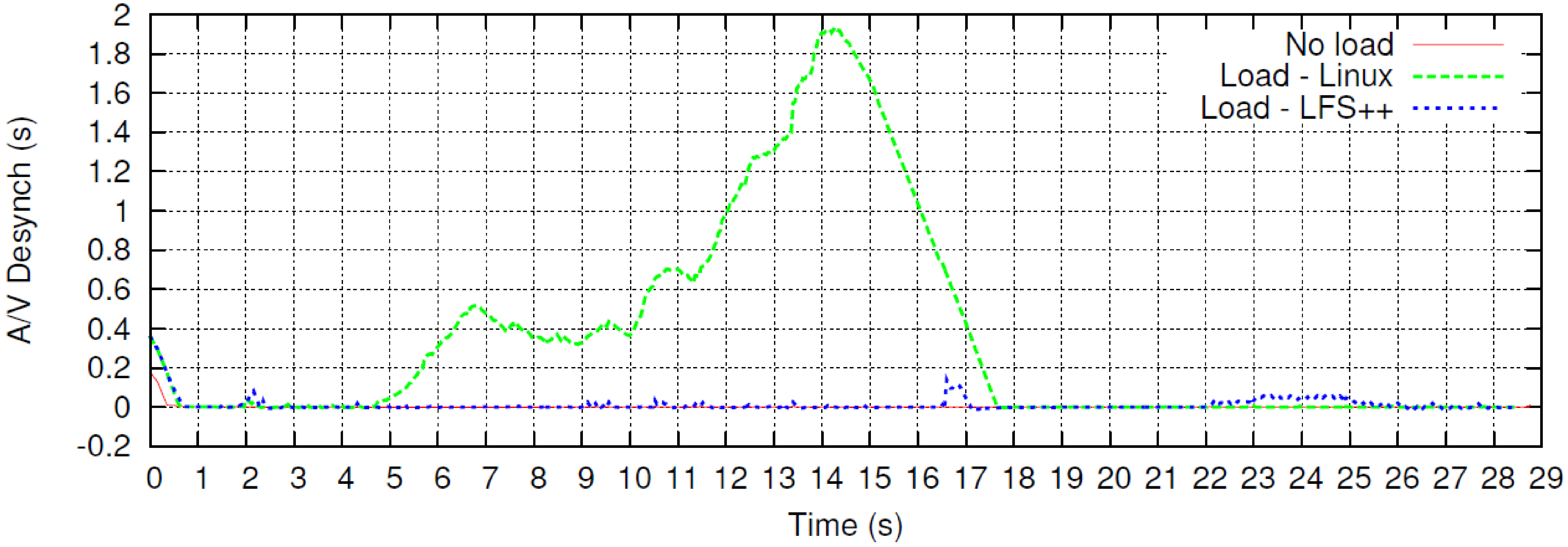


FOURIER

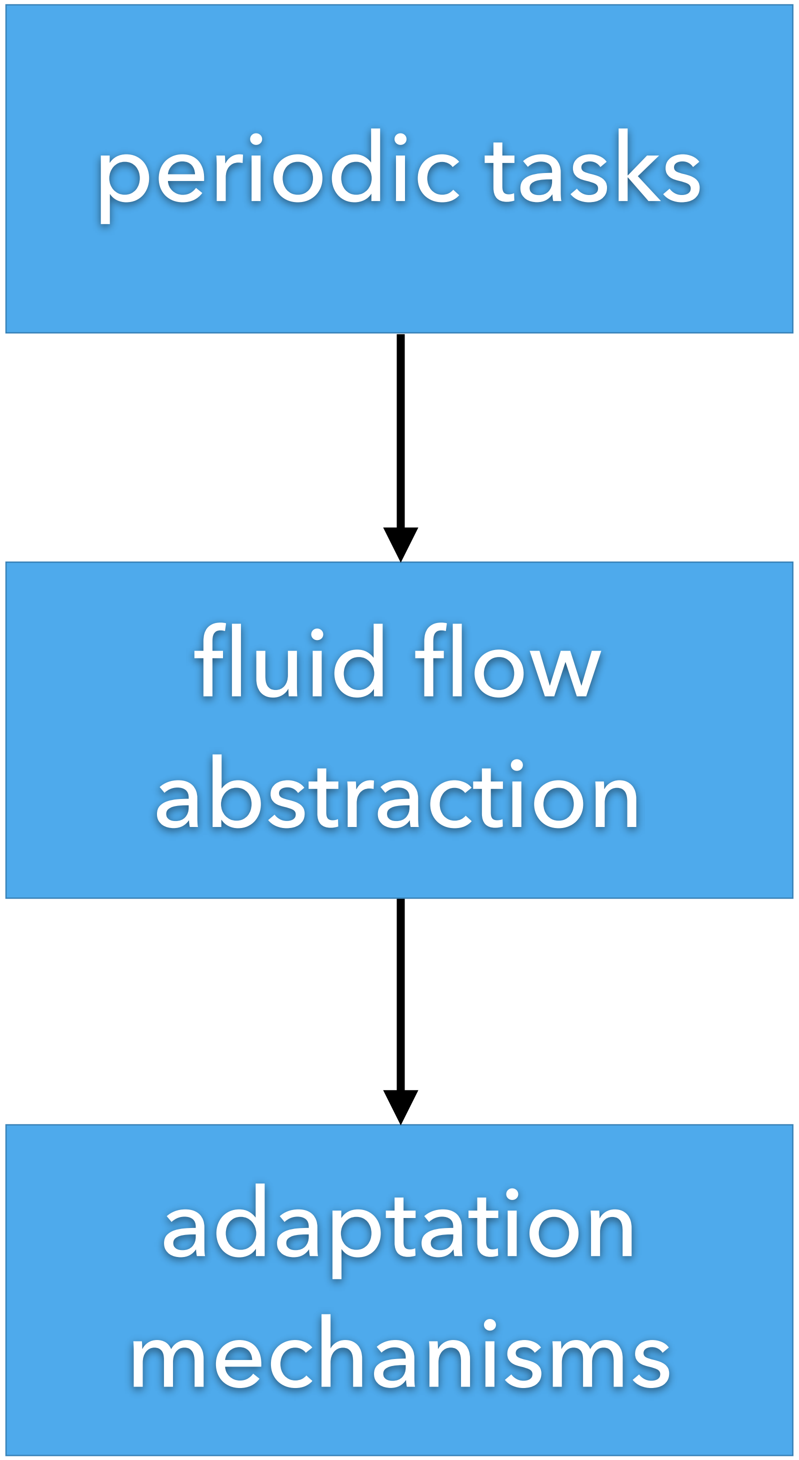

0
55120
55160
55200
5524
Time (ms)
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
'transform.dat'
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
110
Frequency (Hz)

RESULTS

mplayer desync while starting Eclipse



SUMMARY



periodic tasks

fluid flow abstraction

adaptation mechanisms

✔ predictable job behavior

✘ inflexible

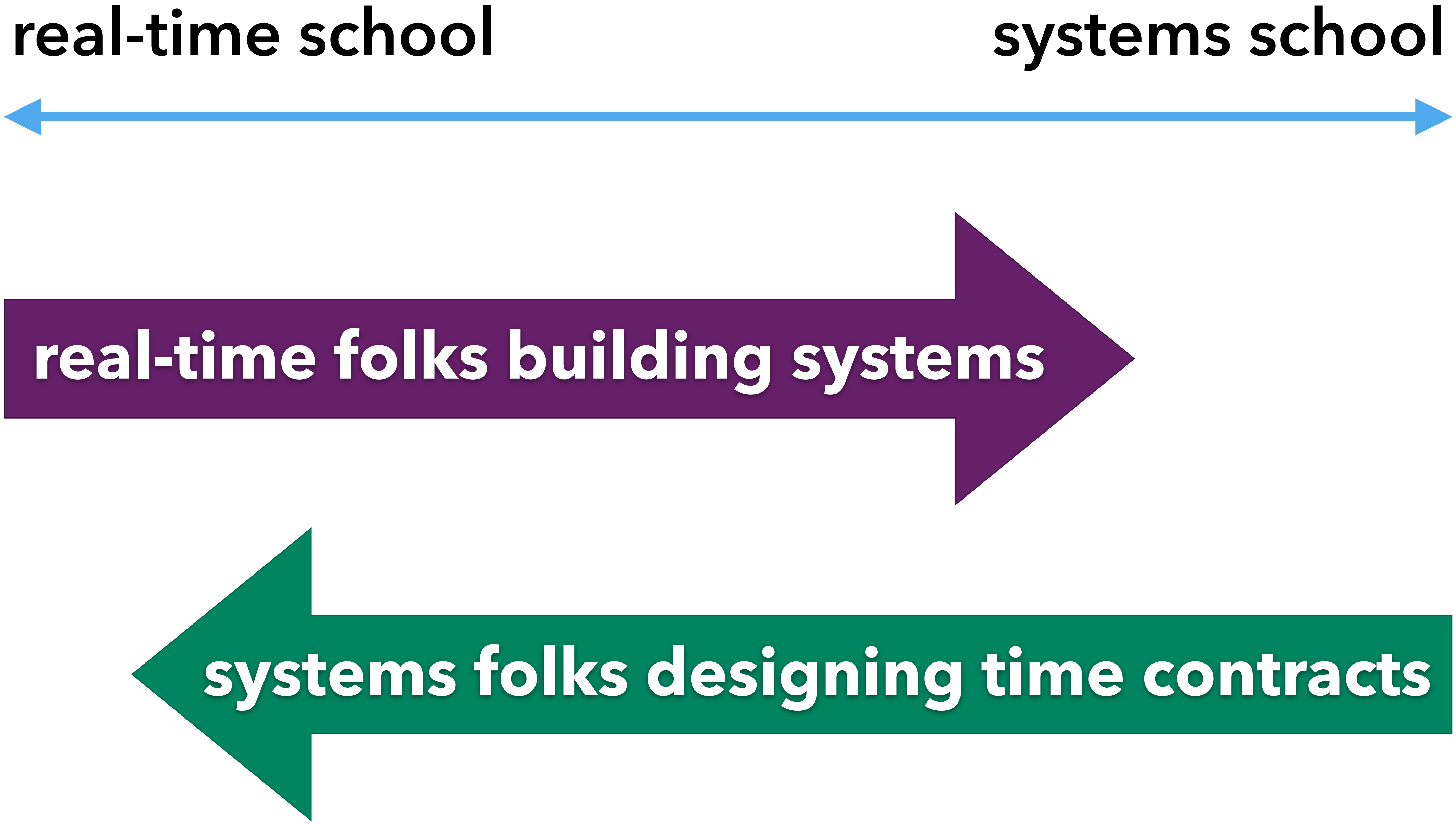
✔ flexible on overruns

✘ job behavior less clear

✔ serve jobs per demand

✘ controller infrastructure

DIRECTIONS



real-time school

systems school

real-time folks building systems

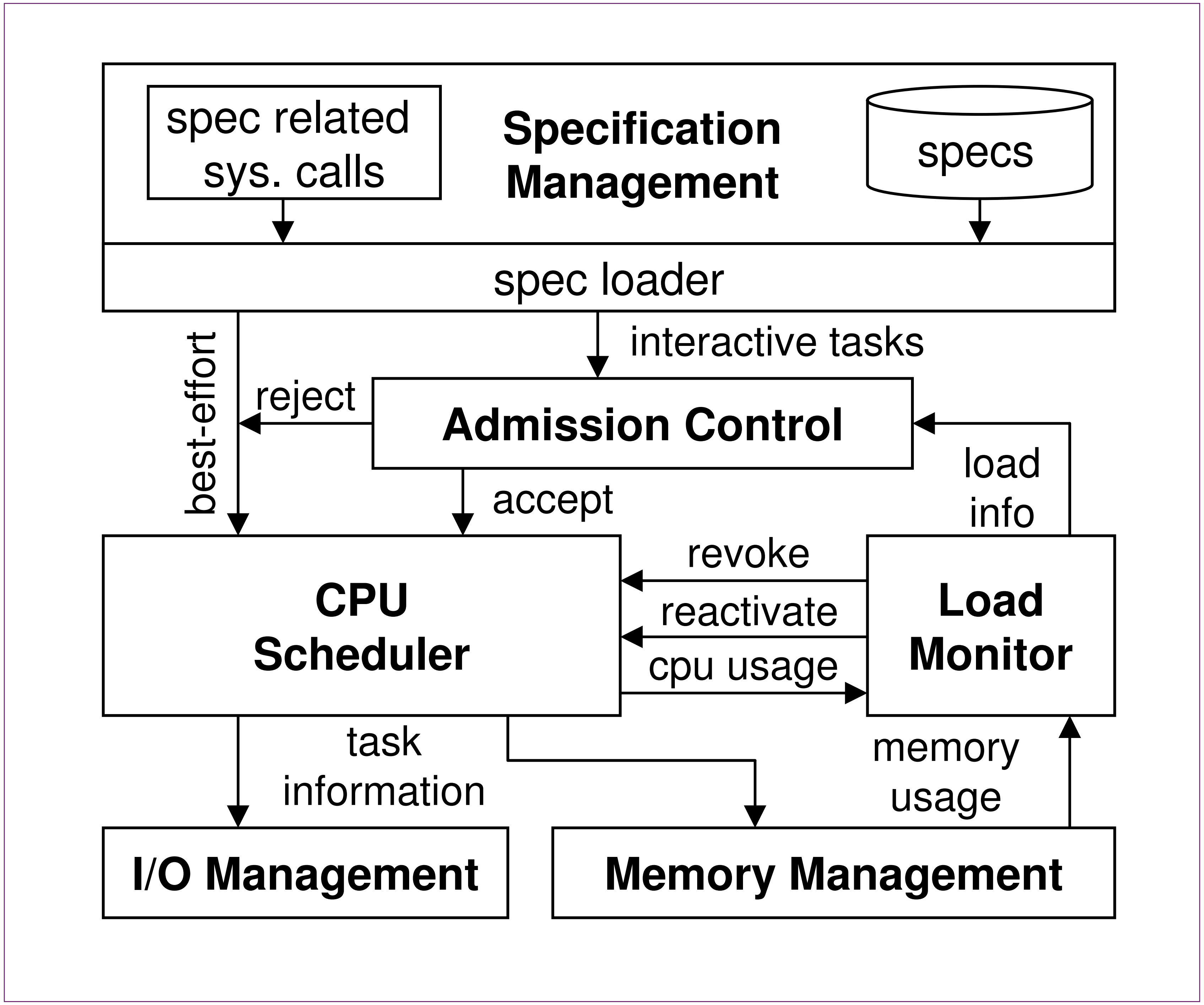
systems folks designing time contracts

REDLINE

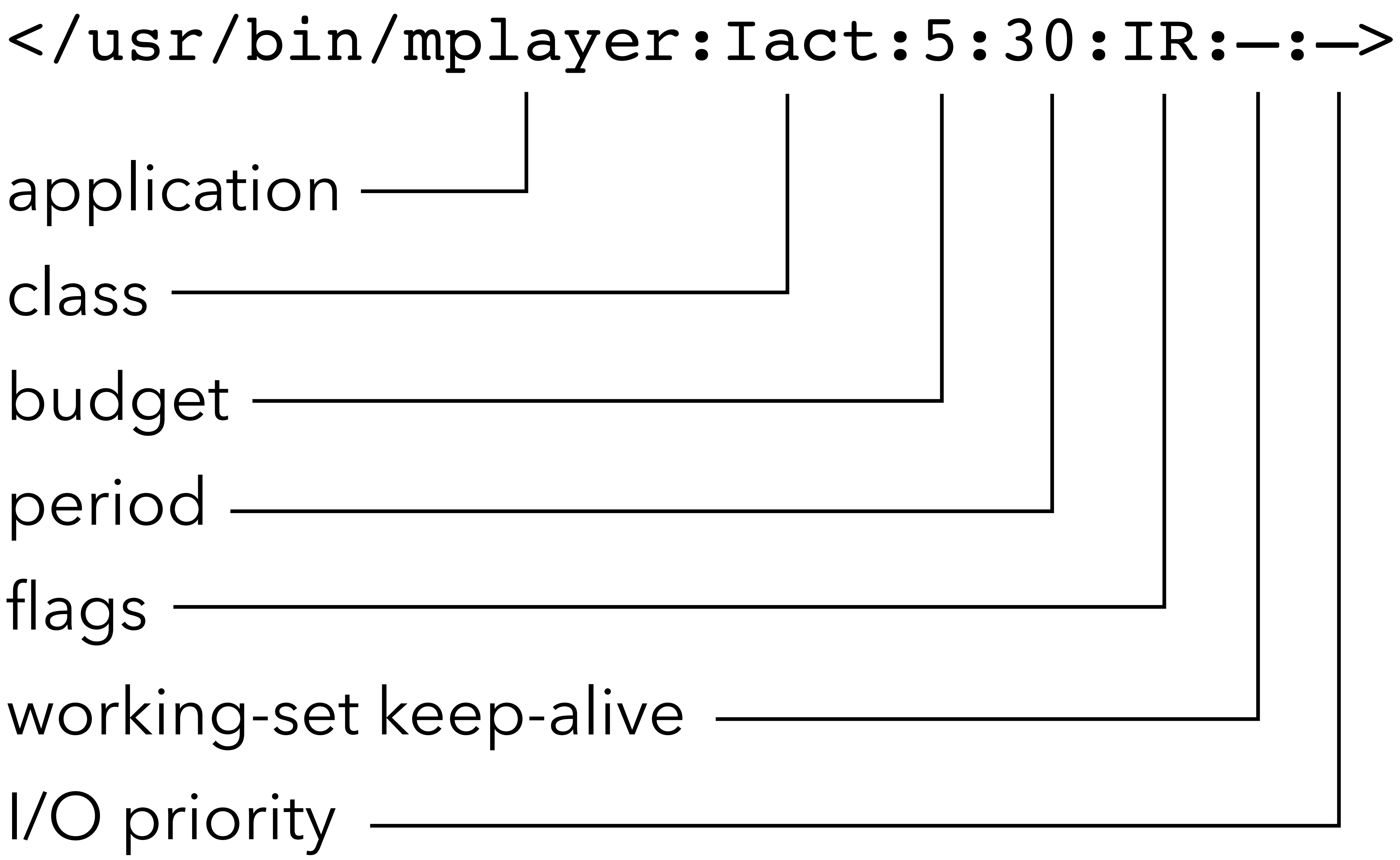
Ting Yang, Tongping Liu, et al.: Redline: First Class Support for Interactivity in Commodity Operating Systems. Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation (OSDI), pp. 73–86, December 2008

- integrated management of CPU, memory and disk to improve responsiveness
- external specification files

REDLINE


spec related
sys. calls
Specification
Management
specs
spec loader
interactive tasks
best-effort
reject
Admission Control
load
info
accept
revoke
CPU
Scheduler
reactivate
Load
Monitor
cpu usage
task
information
memory
usage
I/O Management
Memory Management

REDLINE



```
</usr/bin/mplayer:Iact:5:30:IR:—:—>
```

application

class

budget

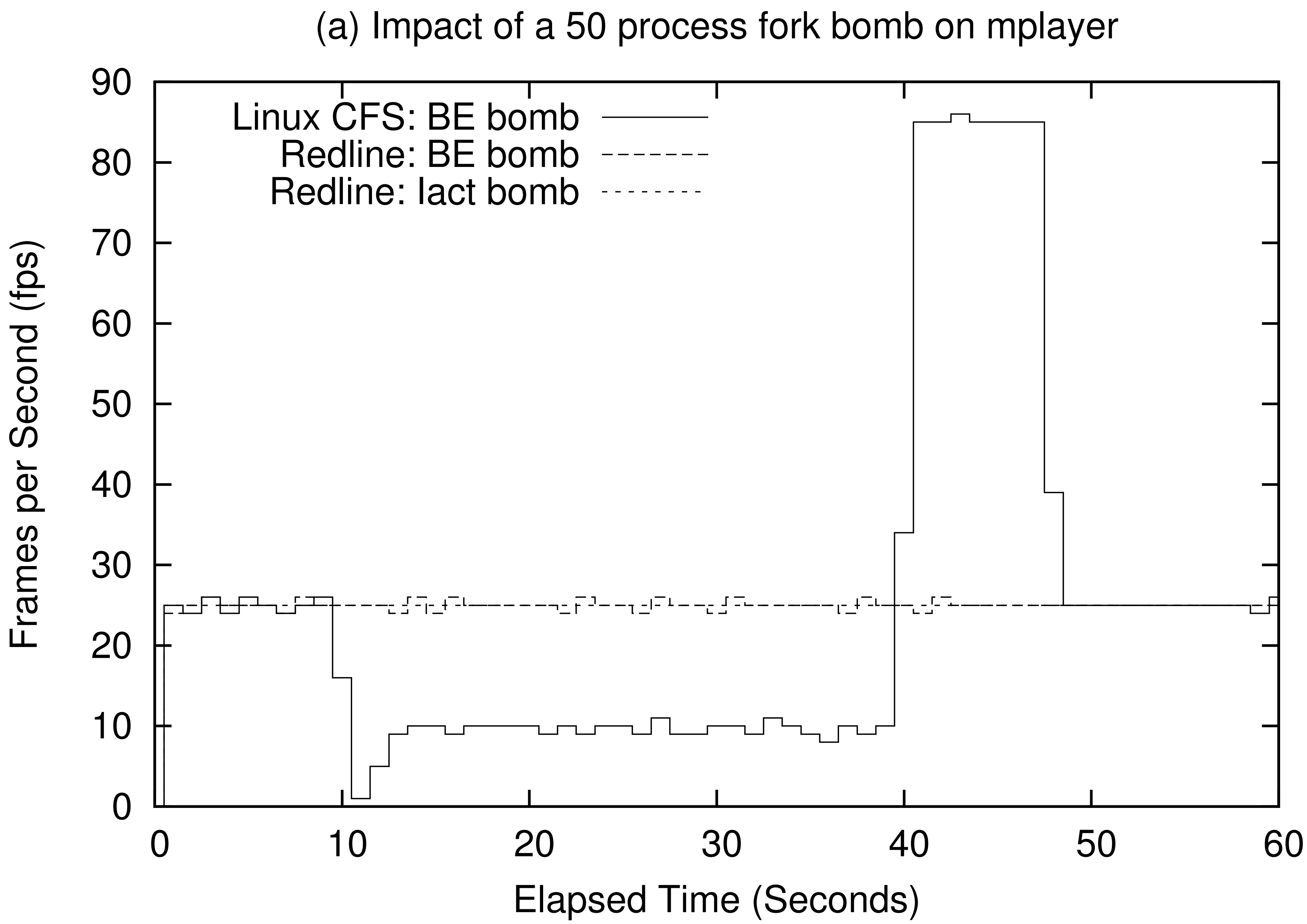
period

flags

working-set keep-alive

I/O priority

REDLINE


(a) Impact of a 50 process fork bomb on mplayer
Linux CFS: BE bomb
Redline: BE bomb
Redline: Iact bomb
Frames per Second (fps)
0
10
20
30
40
50
60
70
80
90
Elapsed Time (Seconds)
0
10
20
30
40
50
60

BVT

Kenneth J. Duda, David R. Cheriton: Borrowed-Virtual-Time (BVT) Scheduling: Supporting Latency-Sensitive Threads in a General-Purpose Scheduler. Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP), pp. 261–276, December 1999

- each thread carries a virtual timestamp
- increases when thread runs, inversely proportional to its weight
- thread with smallest timestamp runs

- this is also the principle behind the Completely Fair Scheduler (CFS) in Linux
- **warp time** controls dispatch latency
- effective virtual time = actual virtual time – warp time
- effective time is used for scheduling
- warping is constrained by other parameters

SUMMARY

- golden middle ground still not found
- a lot of RT-research moves deep into theory territory
- current chance to close the gap: multicore scheduling