Welcome to course 24071

System Architecture

Bellosa, Kupferschmied, Liefländer

October 20 2008

WT 2008/09

System Architecture Group

System Architecture

1 Introduction

Gerd Liefländer

Agenda

- Course
  - Introduction
  - Organization
  - Philosophy
  - TODOs
- First View on System Architecture
- Motivation by Example

Comment: *Slides are in English for* ***2 reasons****:*

- *Previously non German speaking staff members*
- *Good training, because you will/should read some innovative or classical research papers (all in English)*

Notes on Copyright

- Several slides are from the following authors offered on the web as course outlines:
  - A. Tanenbaum: Modern Operating Systems
  - R. Brause: Betriebssysteme, J. W. Goethe Universität, Frankfurt a. M.
  - G. Heiser: Operating Systems, UNSW, Sydney
  - H.U. Heiss: Operating Systems, TU Berlin
  - W. Schröder-Preikschat: Softwaresysteme I, Uni Erlangen
  - A. Silberschatz: Operating System Concepts
  - W. Stallings: Operating Systems
  - M. Welsh: Operating Systems, Harvard University
- Thanks to each of them
- Feel free to use our slides in the same way

Organization

Staff

Course Sites

Intended Schedule

Tutorials

Assignments

Examination

Forum

Other Courses

Lecturers

Lectures:

Frank Bellosa	
E-mail:	bellosa@ira.uka.de
Phone:	608-3834
Meeting Times:	Tuesday: 14:00 – 16:00
Office:	158, 1. Floor, Informatik-Neubau (50.34)

Gerd Liefländer	
E-mail:	lief@ira.uka.de
Phone:	608-3837
Meeting Times:	Tuesday, 15:45 – 17.15
Office:	160, 1st floor, Informatik-Neubau (50.34)

Staff for Lectures & Assignments

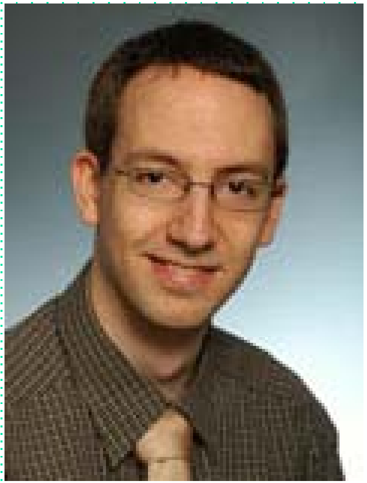
Philipp Kupferschmied

Phone:	0721-608-3836
Email:	**pkupfer@ibds.uka.de**
Meeting Times:	Wednesday, 15:30 – 17:00
Office:	163, 1st floor, Informatik-Neubau

- Whenever there is an open question in one the assignments mail to him
- But please, first read the text of the assignment carefully

Additional Staff Members (L4/PM)

Jan



Raphael

Andreas

Simon

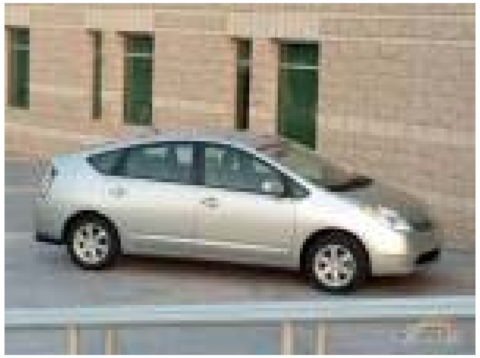
Microkernel
Virtual Machines
HW/OS Co-Design

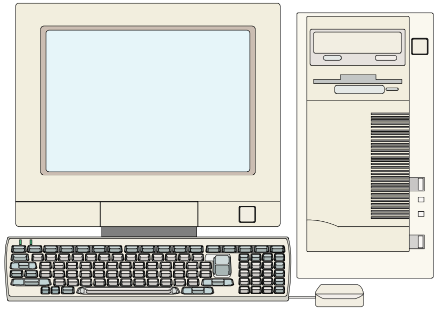
Energy Aware Scheduling
Sensor Networks

Why System Architecture?

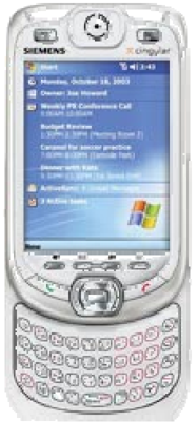
- Some OSes still suffer from design failures of the past
    - OS crashes still happen regularly
    - Often system bugs are due to compatibility requirements
    - To preserve a clean structure of a system is very hard
- Only few of you will ever write an OS from scratch
    - Some of you might be involved in improving an existing OS (e.g. Linux, Free BSD, ...)
    - Some will design and implement an application system ⇒ similar problems to solve as in OS
- In order to do your job well you need to understand the big picture

Computing Devices Everywhere





All these "computers" need properly designed application and operating systems



Famous Quotes

- “I think there is a world market for maybe five computers”
  - Thomas Watson, IBM, 1943
- “There is no reason for any individual to have a computer in their home”
  - Ken Olson, Digital, 1977
- “640KB RAM ought to be enough for anybody”
  - Bill Gates, Microsoft, 1981

Course Page

`http://i30www.ira.uka.de/teaching/`

- “Everything” is on the Web
  - Lecture notes in German xyz.html
  - Course slides in xyz.pps and xyz.pd with additional information on the note parts of the slides
  - Assignments and Solutions (System Architecture (Tutorials))
  - Literature
  - Additional
  - Recommended links
  - Forum
  - Past examinations

Supporting People

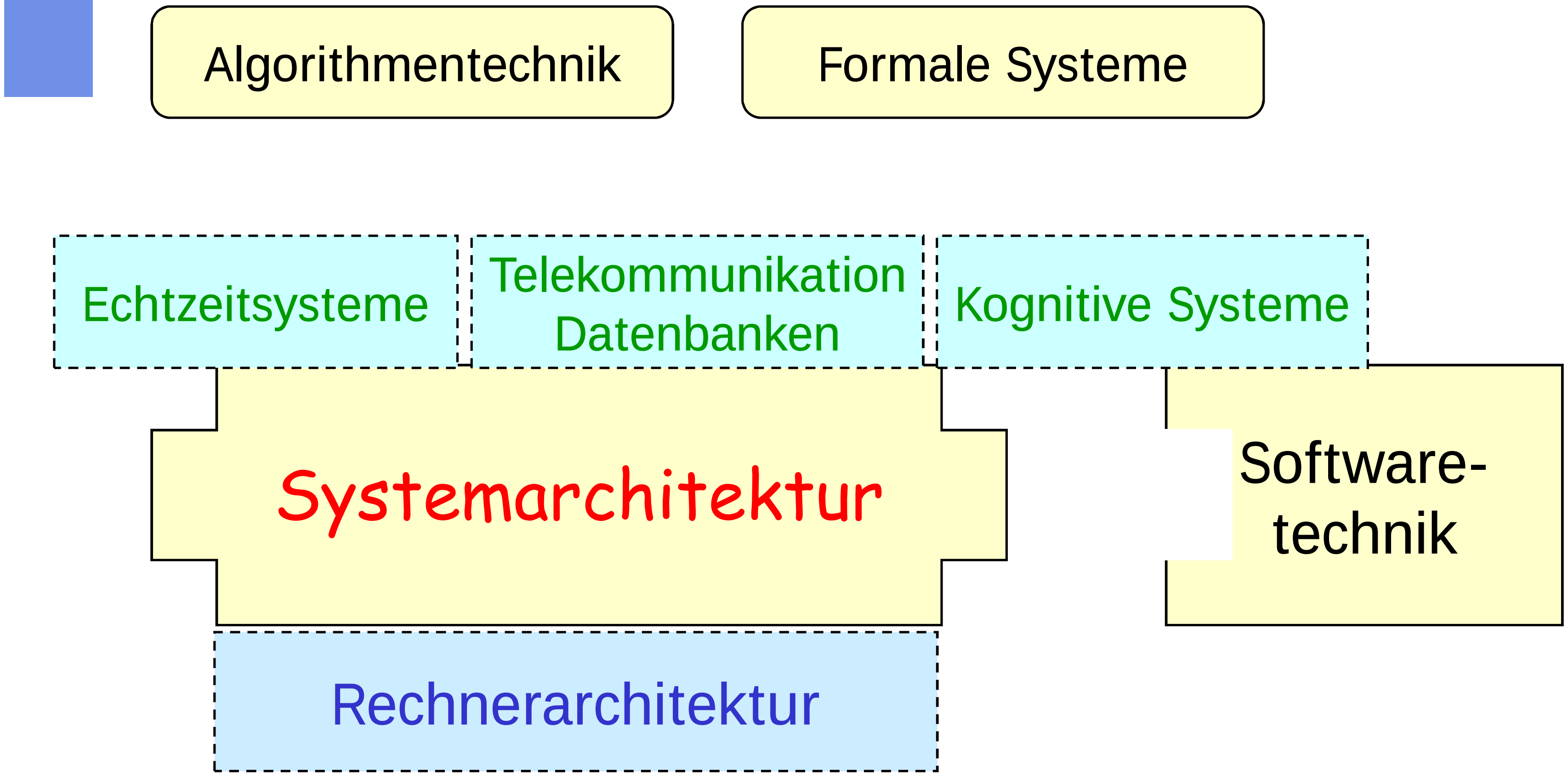
Webmaster	Heinz Zoller
E-mail:	zoller@ira.uka.de
Phone:	608-4045
Office:	150, 1st floor, Informatik-Neubau (50.34)

Secretary	Andrea Engelhart
E-mail:	engelhart@ira.uka.de
Phone:	608-3834
Office:	159, 1st floor, Informatik-Neubau (50.34)

Course Preliminaries

- You should know your
  - Data structures
  - Algorithms
  - Programming
- You should know the basics of HW
  - Performance numbers
    - Capacity of memory
    - Speed of CPU & devices
  - Interface, functionality & characteristics
  - Basic terms, e.g. exceptions versus interrupts

Sys. Architecture & Other Courses



Algorithmentechnik

Formale Systeme

Echtzeitsysteme

Telekommunikation
Datenbanken

Kognitive Systeme

Systemarchitektur

Software-
technik

Rechnerarchitektur

30 Lectures in HSaFSG

Nusselt Hörsaal

Each Monday, 9:45

Each Wednesday, 14:00

Mo 20.10.	Mo 27.10.	Mo 3.11.	**Mo 10.11.**	Mo 17.11.	Mo 24.11.
We 22.10.	We 29.10.	We 5.11.	We 12.11.	We 19.11.	We 26.11.
Mo 1.12.	Mo 8.12.	Mo 15.12.	Mo 22.12.	Mo29.12.	Mo 5.01.
We 3.12.	We 10.12.	We 17.12.	We 24.12.	We 31.12.	We 7.01.
Mo 12.01.	Mo 19.01.	Mo 26.01.	Mo 02.02.	Mo 9.02.	
We 14.01.	We 21.01.	We 28.01.	We 04.02.	We 11.02.	

Christmas

Phillip

Frank

Hot Systems
& System Examples

Assignments (*Übungsblätter*)

- 14 **+ 1** non-programming assignments:
  - Abstract discussion exercises
  - Problem solving exercises
- 4 voluntary programming assignments (PAs)
  - Designing, coding, and testing
- Assignments (& solutions*) published at **http://i30www.ira.uka.de/teaching/**
- Assignment 1 is already published, to be discussed in the tutorials this week

*some weeks later!!!

Why Assignments?

Assignments will

- help you to understand ("learning by doing")
- train you to reason and discuss appropriately and scientifically (a good test for the final)
- Solve the questions in a team
- Vivid tutorials need motivated students, i.e. just attending ≠ participating

Our Courses in WT 2008/09

- Base Lab: System-Architecture (4, Frank, Gerd, Philipp)

- Power Management (2, Frank)
- Advanced Lab: Power Management (2, Andreas, Frank)
- Advanced Systems Seminar:

  "Persisten Operating Systems"

  (2, Frank, Philipp, Raphael)

Courses in ST 2009

- Base Lab: System-Architecture (4, Gerd)

- System Design and Implementation (2 + 2, Jan)
- Distributed Operating Systems (3+1, Gerd)
- Microkernel Construction (2, Raphael)
- Real Time Scheduling (2, Claude Hamann)

Analogy to Architecture

Good and Bad Examples

Education of Architects

Famous Real Architectures

- Pyramids of Egypt (or Central America)
- Hanging Gardens of Babylon
- Acropolis in Athens
- Colosseum in Rome
- Taj Mahal in India
- Cathedral Notre Dame in Paris
- House of Parliament in London
- Chrysler Building in New York
- Allianz Arena in Munich

- and other marvelous architectures around the globe

Pyramids of Gizeh*

*One of the seven ancient miracles of the world

Bad Examples

Tower of Babylon (Sign of Megalomania)

Bad Examples

Tower of Babylon (Sign of Megalomania)

Cathedral of Beauvais (Partial Collapse)

Bad Examples

Tower of Babylon (Sign of Megalomania)

Cathedral of Beauvais (Partial Collapse)

Leaning Tower of Pisa (Built on Sand)

Bad Examples

Tower of Babylon (Sign of Megalomania)

Cathedral of Beauvais (Partial Collapse)

Leaning Tower of Pisa (Built on Sand)

Tacoma Bridge (Too much Wind → Free Oscillation)

Bad Examples

Tower of Babylon (Sign of Megalomania)

Cathedral of Beauvais (Partial Collapse)

Leaning Tower of Pisa (Built on Sand)

Tacoma Bridge (Too much Wind → Free Oscillation)

Egyptian Bridge
(Wilhelm von Traitteur from Karlsruhe)

St. Petersburg

Soldiers in uniform step

20. January 1905

Bad Examples

Tower of Babylon (Sign of Megalomania)

Cathedral of Beauvais (Partial Collapse)

Leaning Tower of Pisa (Built on Sand)

Tacoma Bridge (Too much Wind → Free Oscillation)

Egyptian Bridge (Soldiers in Uniform Step)

Former Congress Hall, now House of Cultures (Bad Ferro-Concrete)
and some others round the globe,
e.g. terminal at C. de Gaulle Airport, Paris

<u>Question:</u> *Have real architects learned their lessons?* Yes and no.

Education of Architects

- History of Architecture
- Design and Modeling
- Static and Construction
- Industrial Production of Components
- ...

Introduction to System Architecture

What is System Architecture?

OS: Our Main Example

Central Point of a System Architecture

General + Specific Goals of this Course

Literature

System Architecture?

System ~ something systematic

There are different kinds of systems, e.g.

- Biological Systems (plants, animals)
- Social Systems (insurance, health)
- Economic Systems (market, stock exchange)
- Technical Systems (aqueduct, engines, TV, etc.)
- Scientific Systems (system of chemical elements)
- Astronomic Systems (galaxy, interstellar cloud)
- Traffic Systems (cars, railways, ships, airplanes, etc.)
- Hardware/Software Systems, e.g.
  - General (multi-purpose) systems
  - Specialized (customized) systems

Concepts

- No architect would ever propose a window as the single perfect entrance to a bungalow ⇒
- Be familiar with all system concepts, with their meaning, and with their exact terms, e.g.
  - resource management ≠ resource driver
- *What might happen otherwise?*
  - Again similar severe system bugs, see:

`http://catless.ncl.ac.uk/Risks/index.1.html`

Some lecturers are not that strict, but we are

System Architecture?

Architecture in general is

- something harmonic & beautiful
- (a)esthetically embedded in its environment

Architecture should also include

- some comfort and usefulness

Architecture is also the discipline itself, e.g.

- how to design a system

System Architecture ?

System Architecture = something

understandable	(for you)
explainable	(for us)
correct	(fulfilling specifications)
structured	(for designers)
modularized	(for programmers)
robust	(for system administrators)
reusable	(for companies)
extensible	(for future applications)
scalable	(for additional components)
efficient	(for users)
secure	(for owners and users)

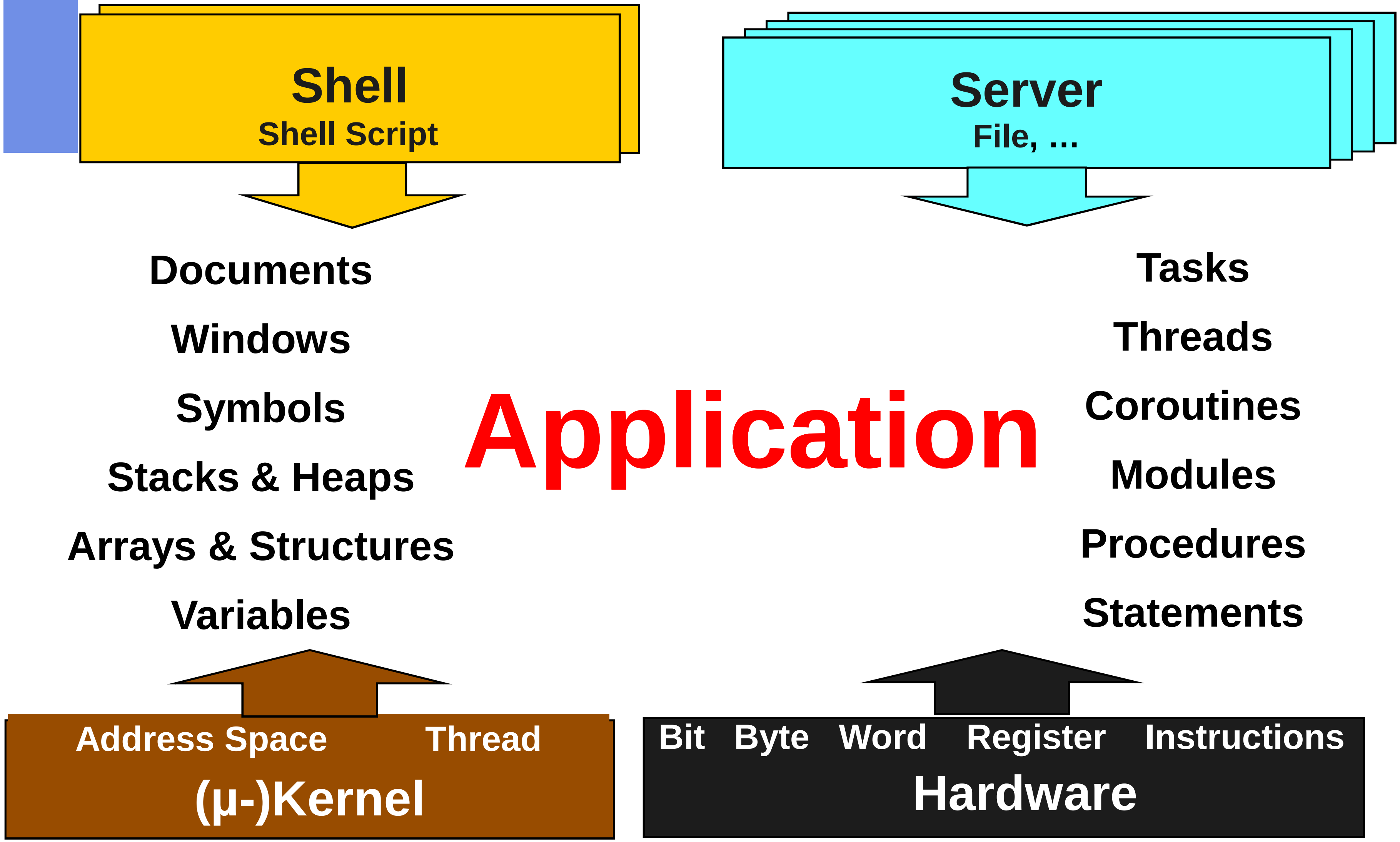
Main Example

Operating System

What is an OS?

- Total sum of all programs controlling and managing the system resources
- Software of a computer that is necessary to run applications efficiently and effectively

Focus of an IT System



Shell

Shell Script

Server

File, …

Documents

Windows

Symbols

Stacks & Heaps

Arrays & Structures

Variables

Application

Tasks

Threads

Coroutines

Modules

Procedures

Statements

Address Space Thread

(μ-)Kernel

Bit Byte Word Register Instructions

Hardware

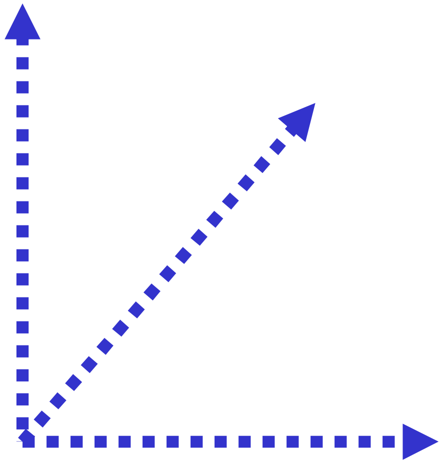
General Goals of an OS Course

- Construct efficient(ly*) small ... large systems
  - Decompose systems into subsystems (objects, modules, components, instances)
  - Understand problems of concurrency and the HW/SW-interaction
  - Distinguish between policies and mechanisms

*topic of the course “software technique”

Specific Goal of this Course

- Decompose a system into well-defined cooperating system components
- Use orthogonal design parameters for the system & for each system component

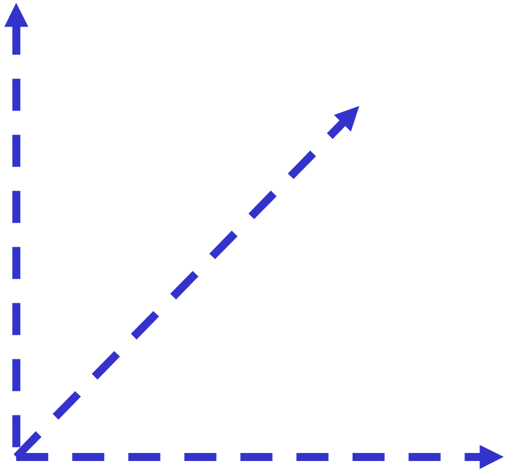


Please, never forget!

Orthogonal Design Parameters

Object = car

- Number of wheels
  - < 4
  - **4**
  - > 4
- Kind of engine
  - **Otto motor**
    - Wankel motor
  - Electric motor
  - Hybrid motor
  - ...
- ...

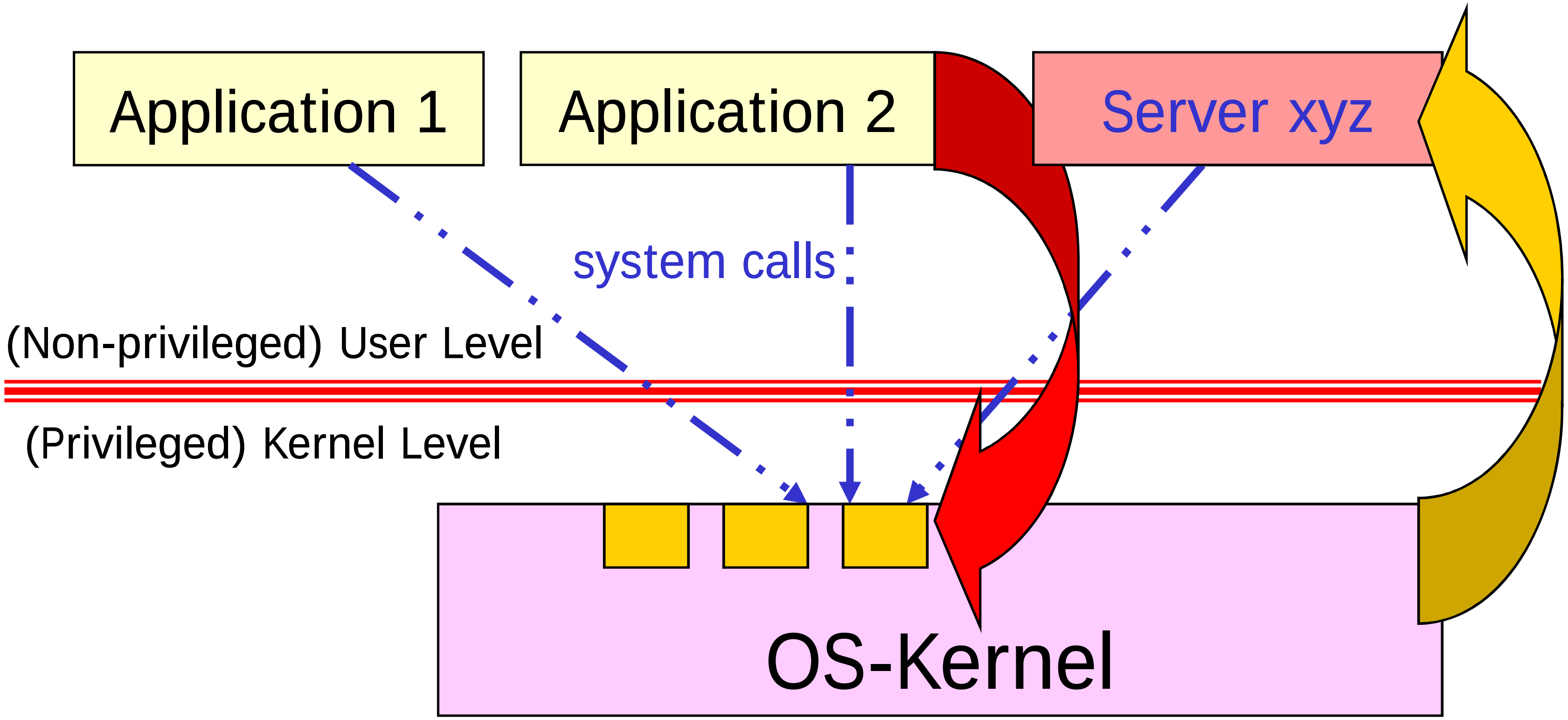


Tentative Course Schedule

- Introduction
- Overview, Motivation, Problems
- Tasks, Processes, Threads
- Thread Switch, Thread States
- Concurrency Problems
- IPC, Deadlock, Starvation
- L4 IPC and Thread Switch
- Scheduling
- Priorities and Resource Contention
- Memory Management
- Address Space Management
- Virtual Memory
- I/O-Management
- File Management, File Systems
- Virtual Machines
- System Examples

Only local systems

OS-Kernel: Privileged Component



2 major trends (starting from monolithic OS-Kernels)

(a) Extensible kernel

(b) Multi-Server on top of a microkernel

OS-Kernel: Privileged Component

- OS-kernel helps to execute different kinds of "applications"
    - Regular applications
        - Emacs
        - Mozilla
        - PowerPoint
        - Games ...
    - Administrative servers
        - cron: runs jobs at pre-scheduled times
        - sshd: manages incoming ssh connections
        - lpd: queues up jobs for the printer

Structure of Application & Server

- Many system activities are encapsulated in the activation model process
  - A process consists of
    - Single execution context: contents of user registers, instruction pointer, status register, stack pointer
    - OS resources: open files, network sockets, ...
    - Address space: (virtual) memory regions for storing program code and data
  - A task can have multiple execution contexts
    - An execution context within a task, i.e. within the same address space, is called a thread

“Kernel” Components

- Process/Task/Thread & Address Management
- Inter-Process Communication (IPC)
- Multiprocessor Management
- Memory Management
- I/O-Management
  - Terminal, modem, keyboard, ...
  - Disk, floppy, CD-ROM, tape, ...
- File System
- Protection
- Accounting
- Networking
  - TCP/IP stack

Only part of a Macro kernel

Not in this lecture

Objectives of an OS

- Abstraction and Standardization
  - Hide HW details
  - Provide uniform interface for different devices
- Generality
  - Changes in characteristics of major applications should not require a complete redesign of the OS
- Customizability
  - Adapt to specific applications' requirements
- Convenience
  - Facilitate programming of applications

Objectives of an OS

- Extensibility
  - Developing, testing and introducing new system functions without interfering with current service
- Scalability
  - System can face increasing load and system components
- Efficiency
  - Use the resources
  - Be fast
  - Consume little power!!! (Important for portables)

Objectives of an OS

- Quality of Service
  - Guarantee certain degree of system service
- Robustness
  - No system crash due to malicious or erroneous applications (or system components)
- Security and Protection
  - Provide authorized access
  - Guarantee privacy and integrity
- Maintainability
  - Main challenge of today’s systems

Job of a System Architect

1. Design useful and customizable systems containing the required functionality
2. Design and structure systems in a way that they can be maintained and extended easily
3. Establish additionally non-functional properties, e.g. quality-of-service guarantees (QoS)
4. Implement correct*, secure, and robust systems
5. ...

*Correct ⇒ according to a previous specification

Literature (1): General Textbooks

Bacon, J.: Operating Systems, Addison-Wesley, 2003

Davis, W.: Operating Systems, Addison-Wesley, 2004

Nehmer, J: Grundlagen moderner Betriebssysteme, dpunkt, 2001

Magee, J.: Concurrency: State Models & Java Programs, John Wiley, 1999

Silberschatz, A.: Operating System Concepts, (**7th** Edition) John Wiley & Sons, 2004

Stallings, W.: Operating Systems, Prentice Hall, 2005

Tanenbaum, A.: Modern Operating Systems, Prentice Hall, 2007

Literature (2): Specific Textbooks

Beck, M.: Linux Kernel Internals, A-W, 1998

Bovet, D.: Understanding the Linux Kernel, O'Reily, 2002

Leffler, S.: Design and Implementation of the 4.3BSD
Unix Operating System, Addison-Wesley, 1996

Mauerer, W.: LINUX-Kernelarchitektur, Hanser, 2004

Mohr, J.: SCO Companion: The essential Guide to Users and
System Administrators, Prentice Hall, 1997

Pham, D.: Multithreaded Programming with Windows NT, P H, 1996

Solomon, D.: Inside Windows NT, Microsoft Press (3nd ed.), 2001

Tanenbaum, A.: Operating Systems: Design and Inplementation,
Minix, P H, 2006

Vahalia, U.: Unix Internals, The New Frontiers, Prentice Hall, 1996

System Information on the Web

- http://www.williamstallings.com/OS4e.html (other slides and student information)
- http://codex.cs.yale.edu/avi/os-book/os7/
- http://joda.cis.temple.edu/courses-os.html (+ many OS courses all over the world)
- http://www4.informatik.uni-erlangen.de/Lehre
- http://wwwagss.informatik.uni-kl.de/Lehre

Motivation by Example

Example: Memory Management

- Initializing a matrix

```
void setA(int *mx, int n, int v)
{
 int i, j;
 for (i=0; i<n; i++)
    for (j=0; j<n; j++)
       mx[i*n+j] = v;
}
```

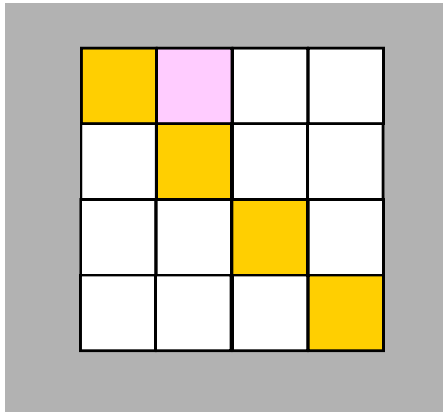
```
void setB(int *mx, int n, int v)
{
 int i, j;
 for (j=0; j<n; j++)
    for (i=0; i<n; i++)
       mx[i*n+j] = v;
}
```

```
mx           pointer to matrix m[n][n] with
             n rows and n columns
mx[i*n+j]    is m[i][j]
```

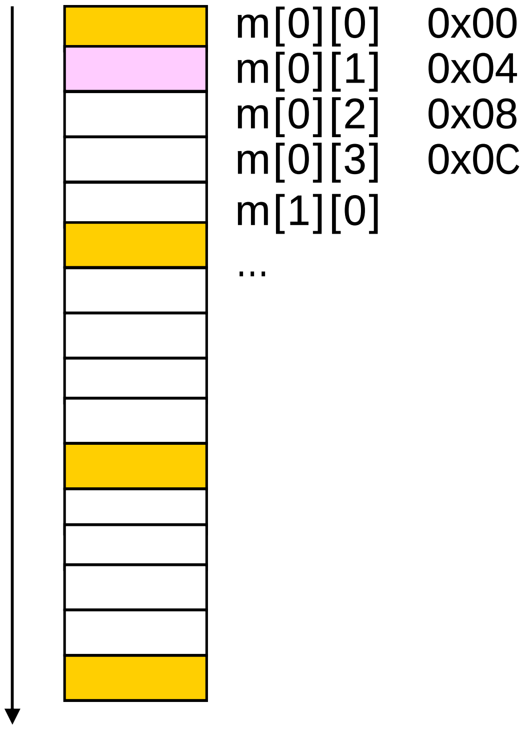
Question: Are there any run time differences?

Mapping of m[i][j] to RAM Cells

abstract view



store by row (as in C/C++)



m[0][0]	0x00
m[0][1]	0x04
m[0][2]	0x08
m[0][3]	0x0C
m[1][0]	
…	

How to access m[0][1] *?*

Thesis:

As long as m[n][n] fits completely in RAM and
∃ uniform memory access no runtime difference, but ...
amount of memory an application gets might be smaller and
∃ additionally caching effects will have influence

Principle of Virtual Memory

- *How to run large applications on a small RAM?*
- Trick:
  - Unused application parts are mapped to disk
  - Mapped to RAM only on demand
  - However, loading on demand is very slow compared to an access to RAM

⇒ Avoid programs involving many long “jumps”

Observation:

`setB()` does not avoid long jumps whereas

`setA()` obeys the principle of locality

Runtime Behavior

- Matrix m[10 000][10 000]

	setA()	setB()
Mac OS X, 0.5 GB RAM 1.25 GHz PPC G4	**user 0.45** system 1.15 total 1.84	**user 17.413** system 2.037 total 20.097
Linux, 4 GB RAM 2x 3 GHz P4 Xeon	**user 0.42** system 0.73 total 1.2	**user 12.25** system 0.733 total 12.99
SunOS, 8 GB RAM 2x 1 GHz Ultra Sparc	**user 1.645** system 0.89 total 2.87	**user 45.495** system 0.885 total 47.725
Windows, 4 GB RAM 2x 3 GHz P4 Xeon	**user 0.843** system 0.25 total 1.125	**user 9.937** system 0.250 total 10.344

Milestones of Systems Research

1956 IBM 704 OS	
1968 THE	Dijkstra (U Eindhoven, Netherlands)
1968 Arpanet	Shapiro, Crocker, Carr, Rulifson, Stoughton (UCLA, NWG)
1969 Multics	Organick, Saltzer et al. (MIT)
1970 RC4000	Brinch Hansen (Kopenhavn, Denmark)
1971 Unix	Ritchie, Thompson (Bell Labs)
1978 VM/370	Seawright, MacKinnon (IBM)
1981 Hydra (OOS)	Wulf (MIT)
1986 Amoeba (DS)	Mullender (CWI; Netherlands)
1989 Mach (μK)	Rashid (CMU)
1990 WWW	Berners-Lee, Cailliau (Cern, Switzerland)

However, we are still looking for an ideal system:

fast, robust, secure, configurable and customizable

Back to the Future?

Interesting breath-taking papers

- Fernando Corbato Turing Award Lecture (TAL): On Building Systems That Will Fail, also in C.ACM, September 1991
- Jim Gray’s TAL: What’s Next? A Dozen Remaining IT-Problems
- Marvin Minski’s TAL: Form and Content
- Marc Smotherman: IBM Advanced Computing Systems, a Secret 1960’s Supercomputer Project
- Ken Thompson’s TAL: Reflections on Trusting Trust
- Butler W. Lampson: Hints for computer system design, 9th SOSP, 1993