

Operating Systems - Introduction

András Millinghoffer

<http://www.mit.bme.hu/~milli>

milli@mit.bme.hu

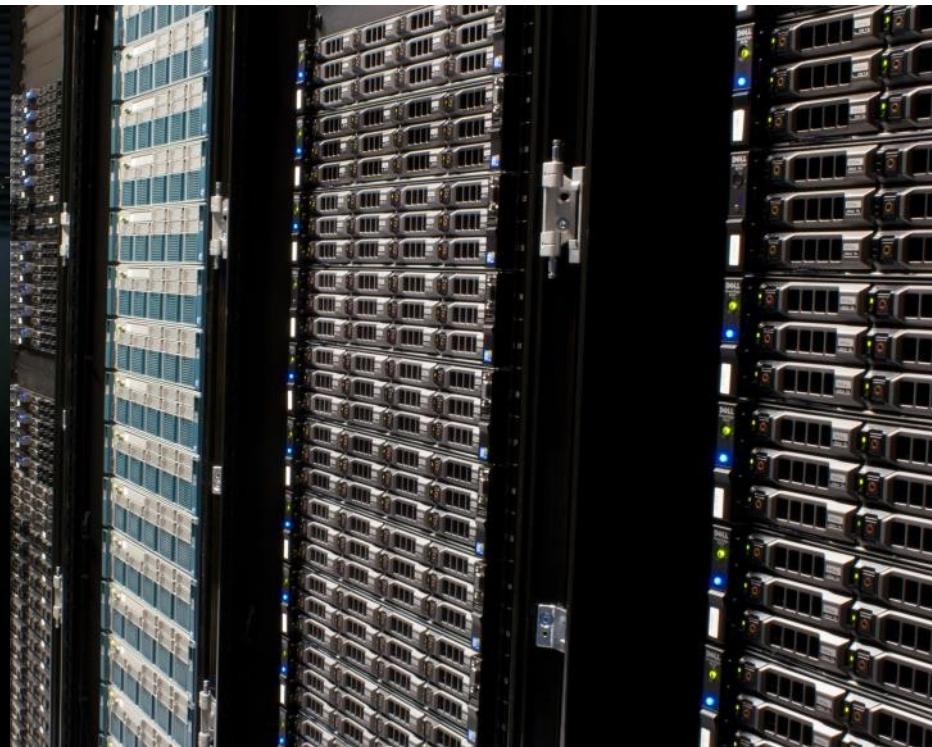
Budapest University of Technology and Economics (BME)
Department of Artificial Intelligence and Systems Engineering (MIT)

The slides of the latest lecture will be on the course page. (<https://www.mit.bme.hu/eng/oktatas/targyak/vimiab00>)
These slides are under copyright.

The lecturers and useful information

- Lecturer
 - András Millinghoffer
 - IE425, milli@mit.bme.hu
 - Interested in machine learning
 - Bioinformatics
- Course coordinator
 - Tamás Mészáros PhD.
 - IE437, meszaros@mit.bme.hu
- Official information on the web
 - <https://www.mit.bme.hu/eng/oktatas/targyak/vimiab00>
 - OR: mit.bme.hu -> english() -> education -> Operating Systems (VIMIAB00)





How complex is an OS?

Windows XP: [45 million LOC](#)

MINIX core < 1400 LOC whole OS < 5000 LOC

Linux 3.1 kernel: 37 000 files, 14 million LOC – 4.17 [23 million LOC](#)

Linux Kernel

**Source Code Visualization
Commit History
June 2, 2012 - Past 6 Weeks**

```
$ git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
```

Music:

Fancy Mike – Death Proof
http://freemusicarchive.org/music/Fancy_Mike/

Rendered/Mixed:

Corey Goldberg
<http://goldb.org>

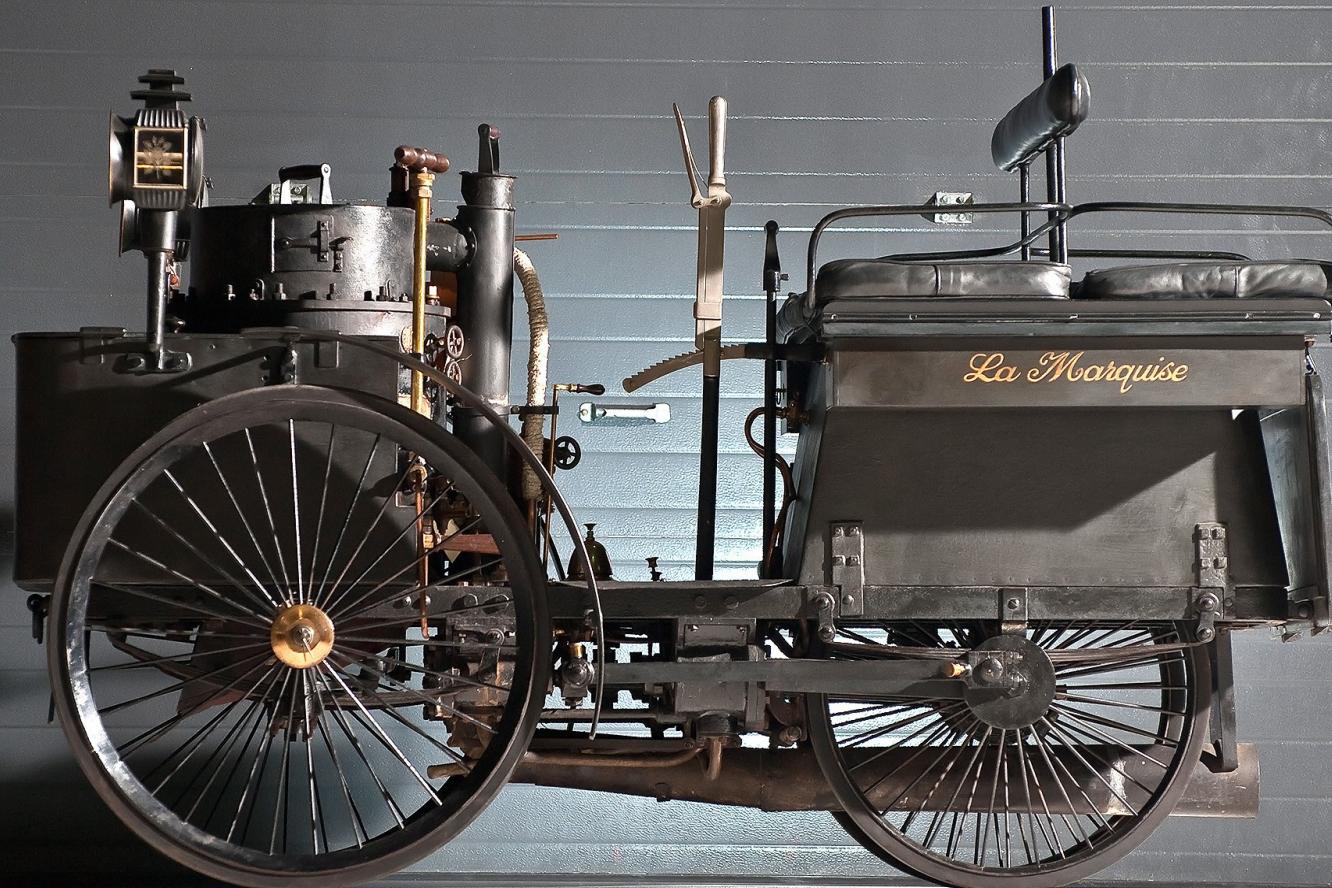




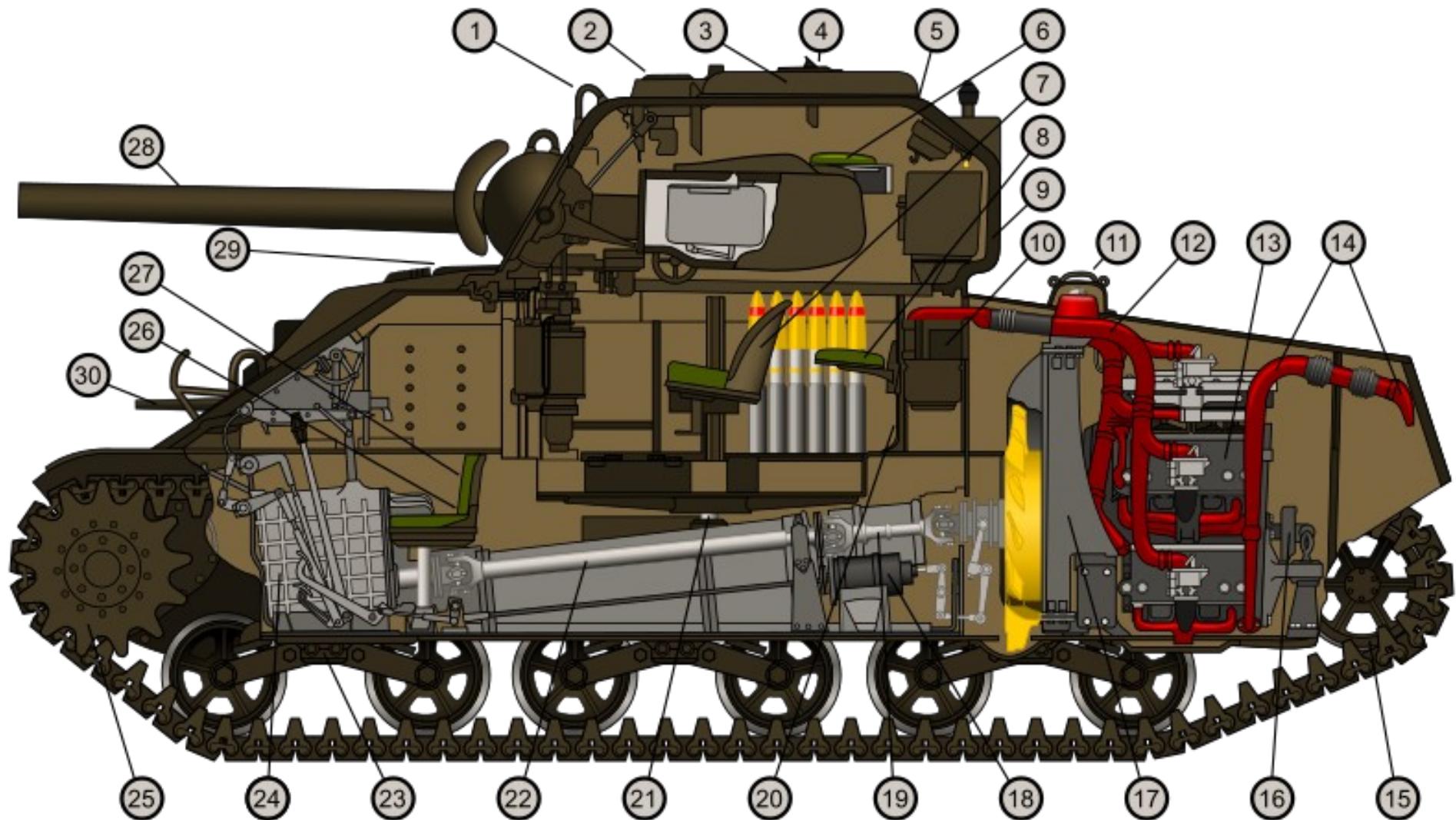
What's the aim of the Operating System (OS)?

- Provides tools for the users
 - users: almost everyone: simple users, developers, system administrators
 - tasks: connections, information retrieval, computations, gaming, storage, ...
also: software development and system administration
 - tools: computer (PC, smartphone...), network, printer, disc, ...
but also the software tools of the OS.
- The OS is an ecosystem for the programs
 - It provides a „living” environment and resources
 - Provides standard interfaces for hardware devices
 - Provides computing capability
 - Provides integrated software tools and services like: security, development and administrative, etc. subsystems
 - Managing the shared resources
 - Controlling the running tasks
 - Parallel execution of the tasks
 - Provides communication channels between task to enhance cooperation

How well developed is an OS?



Source: Wikipédia



Source: Wikipédia

Goals of this course

- Learning the OS architecture and operation
 - Architecture (layered, monolithic, microkernel, etc.)
 - Running tasks (process, thread, scheduling, life-cycle, monitoring)
 - Virtual machine abstraction
 - Memory management (physical and virtual memory, paging and swapping)
 - Inter-process communication (FIFO, message queues, shared memory, RPC, socket)
 - Storage systems (files; local, network and distributed file systems)
 - Authentication, authorization and security
- Introducing these specific part in laboratory sessions
 - Windows administration
 - Linux basics: app and service installation, management
 - Embedded real-time operating system: FreeRTOS @ Silabs STK3700
- Encouraging the self-learning
 - This course isn't pure theoretical, this knowledge can be used in practice.

Structure of this course

- Lecture (3 hours a week)
 - Every Monday from 10:15 (QBF09)
 - Every second Tuesday, odd weeks from 14:15 (QBF09)
- Laboratories
 - Every second Tuesday, even weeks from 14:15 (???)
 - 2 topics, a passing mark is required from both
- Mid-term test at the end of semester (OK entry test and min. 40%)
 - Theoretical and practical questions, problem solving
 - Definitions, entry test (not present in midterm exam)
 - basic level of knowledge, mainly definitions
 - Yes-no questions (20 points)
 - general level knowledge
 - correct answers 1 point, wrong answers -0,5 point
 - Short answers (15 points)
 - general level knowledge
 - practical problem to solve (15 points)
 - can you choose and run algorithms?
- *Home practice*
 - *Small tasks to solve in a virtual environment, test, demonstrations*

Laboratories

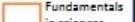
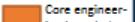
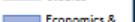
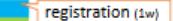
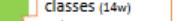
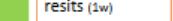
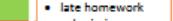
- Laboratory session topics
 - Windows
 - Linux
- Dates To Be Determined
 - Requirements for laboratories
 - Rating
 - Not completed (failed entry test, failed to solve the tasks)
 - Completed (OK for entry test and for the tasks)
- Further details will be posted later

Connection with other courses

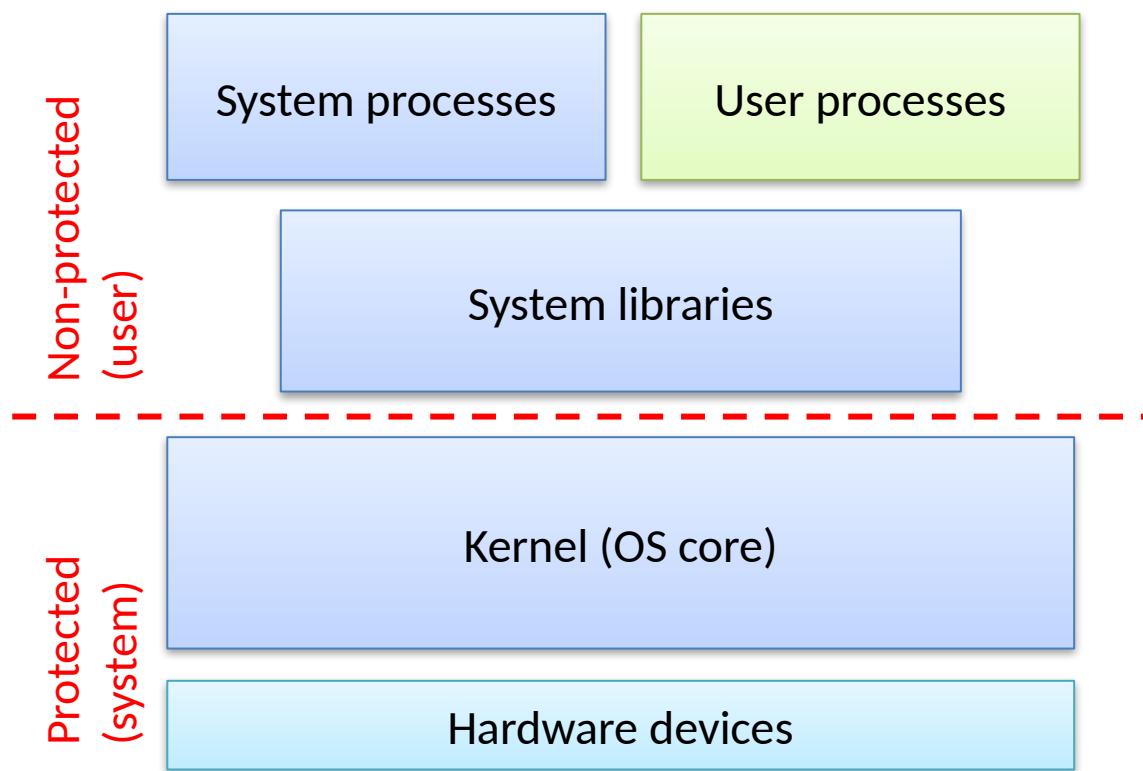
Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics

BSc degree program in Engineering Information Technology

7 semesters, 210 credits
valid from 2014 Fall

BSc degree program in Engineering Information Technology										
7 semesters, 210 credits valid from 2014 Fall										
1 st semester 30 credits 28 h/week	Calculus 1 for informaticians 4/2/0/e/6 BMETE90AX21	Physics 1i 3/1/0/e/4 BMETE11AX23	Introduction to the theory of computing 1 2/2/0/e/4 BMEVIZSAA00	Digital design 3/1/2/e/7 BMEVIMIA01	Basics of programming 1 2/2/2/m/7 BMEVIEEA00	Becoming an engineer 2/0/0/m/2 BMEGT52A400	Legend weekly contact hours (lectures / classroom practice / lab. practice) course title credit value according to ECTS – 1 credit represents 30 work hours 3x BM courseCode requirement m – mid-semester mark e – exam number of similar subjects subject code (as in the Neptun course management)			
2 nd semester 30 credits 27 h/week	Calculus 2 for informaticians 4/2/0/m/6 BMETE90AX22	Physics 2i 3/1/0/e/4 BMETE11AX24	Introduction to the theory of computing 2 2/2/0/e/4 BMEVIZSAA01	System modeling 2/1/0/m/4 BMEVIMIA00	Basics of programming 2 2/2/2/m/7 BMEVIIIA00	Computer architectures 3/1/0/e/5 BMEVIIIA00				
3 rd semester 30 credits 26 h/week	Probability theory 2/2/0/e/4 BMEVIZSAB00	Coding technology 3/0/0/e/4 BMEVIHAB00	Databases 3/1/0/e/5 BMEVITMAB00	Communication networks 1 3/0/1/m/4 BMEVIIHAB01	Basics of programming 3 2/2/2/m/5 BMEVIIIA00	Software engineering 3/0/0/e/4 BMEVIIIA01	System theory 3/1/0/m/4 BMEVIIHAB00			
4 th semester 30 credits 27 h/week	Theory of algorithms 2/2/0/e/4 BMEVIZSAB01	Operating systems 3/0/1/e/5 BMEVIMIA00	Databases laboratory 0/0/2/m/2 BMEVITMAB02	Communication networks 2 3/0/1/e/4 BMEVITMAB01	Software techniques 2/0/2/e/5 BMEVIAUAB00	Software project laboratory 0/0/2/m/2 BMEVIIIA02	Computer graphics 3/0/0/m/4 BMEVIIIA03			
5 th semester 30 credits 27 h/week	Technology of IT devices 3/0/1/m/4 BMEVIEEAC00	Mobile- and web-based software 3/0/1/e/5 BMEVIAUAC00	Artificial intelligence 3/0/0/m/4 BMEVIMIA00	Study specialization subject 3/1/0/e/4 2x	Design laboratory 0/0/2/m/3	Micro- and macroeconomics 4/0/0/e/4 BMEGT30A001	Business law 2/0/0/m/2 BMEGT55A001			
6 th semester 30 credits 29 h/week	IT security 3/0/0/m/3 BMEVIIHAC01	Management of information systems 3/0/1/m/4 BMEVITMAC02	Study specialization subject 3/1/0/e/4 2x	Study specialization laboratory 1 0/0/2/m/2	Project laboratory 0/0/4/m/5	Free elective 4/0/0/e/4	Mandatory human & economic science elective 2/0/0/m/2 2x BMEGT*****			
7 th semester 30 credits 25 h/week	Embedded information systems 3/0/0/m/3 BMEVIMIA00	Study specialization laboratory 2 0/0/2/m/2	BSc thesis work 0/10/0/m/15	Free elective 2/0/0/m/2 3x	Mandatory human & economic science elective 2/0/0/m/2 2x BMEGT*****	THESIS DEFENSE SESSION Organized in the last exam period in front of a committee. Includes presentation of thesis work, discussions and oral exam in one or two subjects. Conditions for admission to the thesis defense session apply.				
Study specialization Topics of the Design lab., Project lab., and BSc thesis work subjects must be related the study specialization block. Enrollment conditions: <ul style="list-style-type: none"> at least 120 credits completed all courses of the first three semesters are completed 		SOFTWARE ENGINEERING Data-driven systems 3/1/0/e/4 BMEVIAUAC01 INFOCOMMUNICATIONS Object-oriented software design 3/1/0/e/4 BMEVIIIA00 Integration & verification techniques 3/1/0/e/4 BMEVIMIA04 Client side technologies 3/1/0/e/4 BMEVIAUAC02 Software development laboratory 1 0/0/2/m/2 BMEVIAUAC03 Software development laboratory 2 0/0/2/m/2 BMEVIAUAD00 Mobile communication networks 3/1/0/e/4 BMEVIIHAC00 Building and operation of networks 3/1/0/e/4 BMEVITMAC00 Media applications & networks in practice 3/1/0/e/4 BMEVITMAC00 Networked resource platforms & apps 3/1/0/e/4 BMEVITMAC03 Infocommunication laboratory 1 0/0/2/m/2 BMEVITMAC03 Infocommunication laboratory 2 0/0/2/m/2 BMEVITMAC03								
		 Fundamentals in sciences  Core engineering knowledge  Specialization studies  Economics & humanities  Free electives								
		 registration (1w)  Classes (14w) <ul style="list-style-type: none"> lectures classroom practices lab. practices quizzes midterms homework assignments  results (1w) <ul style="list-style-type: none"> midterm retakes late homework submission early exams  RESTRICTIONS APPLY								

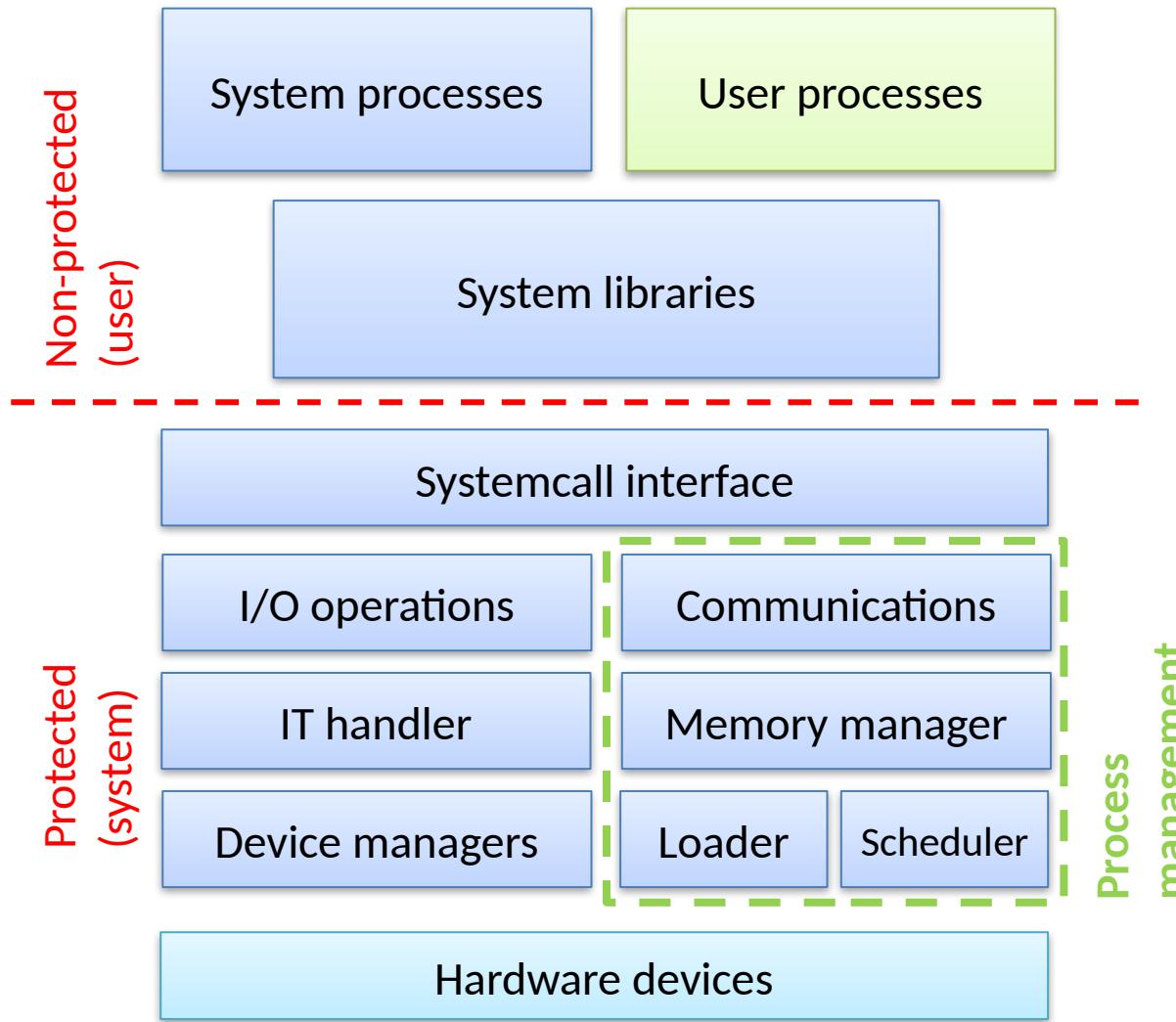
The main blocks of the OS



Kernel: the core (base) of the OS

- Serving user (and system) processes
 - Life cycle monitoring (creation, operation, termination)
 - Providing resources: computational and storage
 - Managing events
 - Providing access to the hardware devices
- Hardware management
 - Managing the accessible physical resources
 - Managing simultaneous queries, separation, solving conflicts
 - Initializing devices
 - Managing and forwarding HW based events
 - Providing a „standard” interface to access these resources
- Controlling reliability, performance and security
 - Protection of the resources from erroneous or noxious usage
 - Separation and protection of the user data
 - Providing common security services

The main blocks of the OS and the kernel



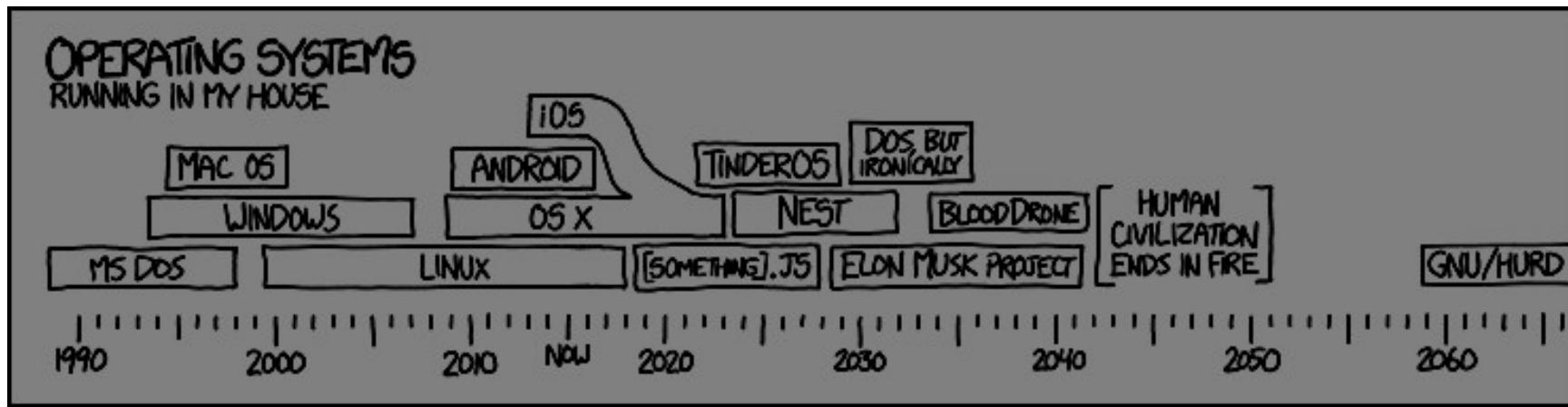
How many lines a kernel?

- Unix kernel
 - Linux 3.1: 37 000 file, 14 million Lines of Code (~half of it is device management)
 - MINIX basic kernel < 1400 LoC, full version: 5000 LoC
- 3D visualization of the Linux kernel
 - <http://www.pabr.org/kernel3d/kernel3d.html>
 - <http://blog.mit.bme.hu/meszaros/node/164>
- Further reading, videos
 - <http://www.jukie.net/bart/blog/linux-kernel-walkthroughs>
 - http://en.wikiversity.org/wiki/Reading_the_Linux_Kernel_Sources

Definition of the OS

- There are many possible definitions
 - Everything that's in the box, when buying an „OS”
 - Sum of the software which makes possible the running of user tasks
 - The program which are allocate the hardware resources
 - The only program which are always running
 -
 - The OS is the sum of the programs which are managing the hardware of the computer and makes possible to run user tasks.
- Do not learn these

Operating Systems: past, present and future



Source: [xkcd](https://xkcd.com/1345/)

Historical overview

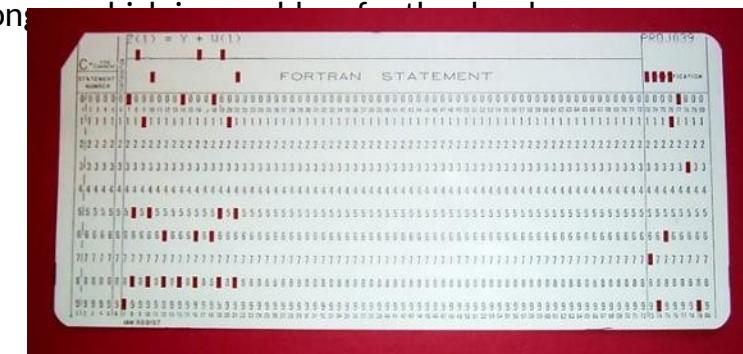


Historical overview

- Related with hardware evolution
 - Appearing new technologies, HW types and services
 - The computing capability is multiplies by 10 every 5 years (slowing)
 - The price/MIPS ratio is multiplied 10 times in 3.5-4 years
- >60 years of experience
 - Hundreds of HW platforms
 - HW, SW combined developement
- Open source codes, many books
- Colorful developement
 - From smartphones to grid servers
 - From engineering/mathematical tasks to today's entertainment devices

Milestones of the OS evolution

- Batch computing – Big room computers
 - The OS is running one task at once
 - The tasks are incoming from punched card, or later magnetic tapes in **batches**.
 - The OS was a device manager and loader
- Multiprogrammed OS (OS/360) – IC, memory, disc
 - The memory is split to sections in order to load more than one process at once
 - If one of processes are waiting for an I/O task, the other one can be run on the processor
 - The running sequence is less predictable and the response time is longer
- Time sharing OS (MULTICS) – HW level protection
 - The processor time is shared equally between users
 - Predictable system and better response times
 - Written in a higher level programming language (PL/I)
- Personal computer OS – LSI circuits, microprocessors,
 - x86
 - Evolution of user interfaces
 - More and more tasks (games, media applications, etc.)
- Embedded OS – Integrated functionality, complex, cheap IC-s
 - Performing real-time tasks in a resource constrained environment



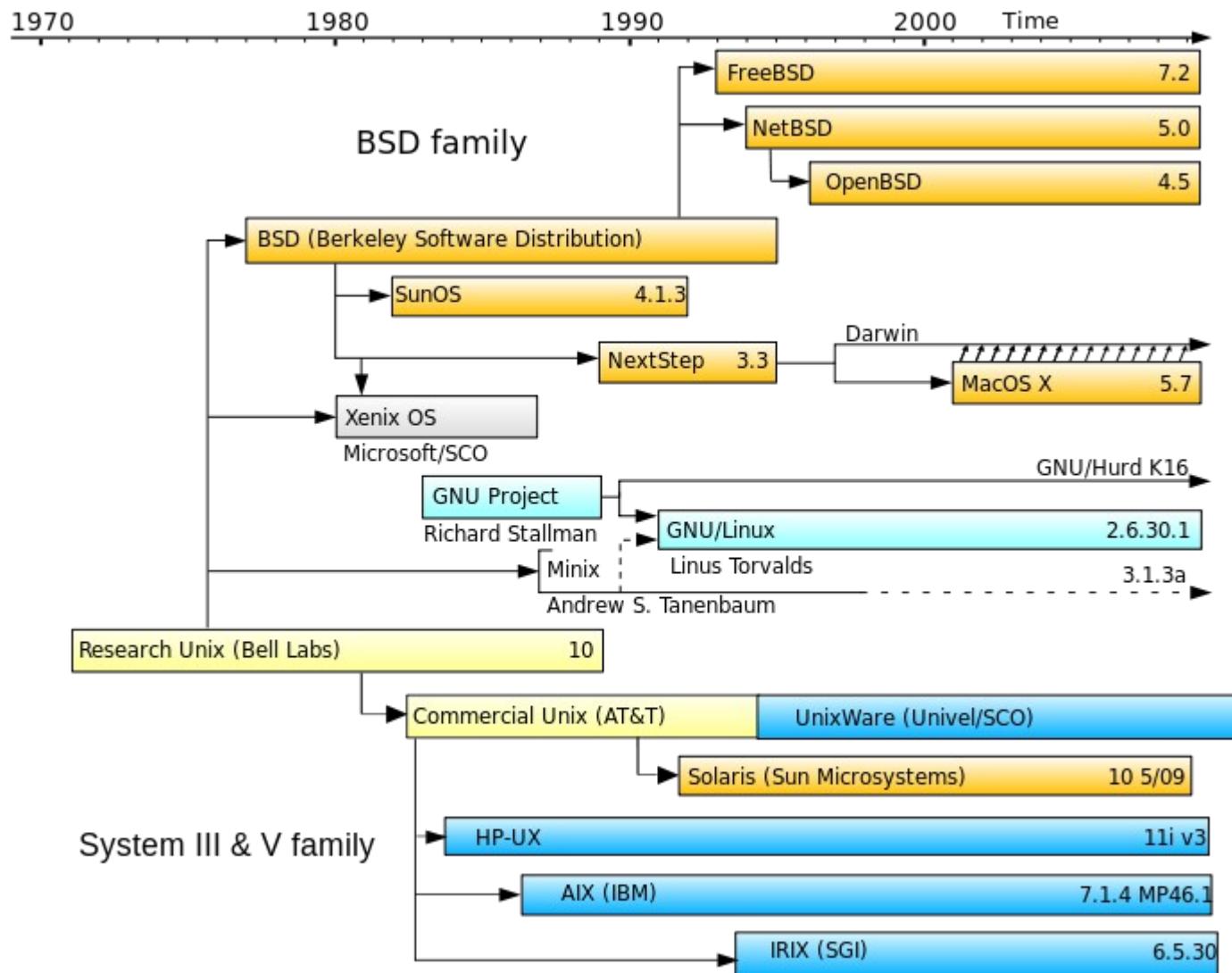
Creation of UNIX

- 1969 AT&T Bell Lab
 - Ken Thompson and Dennis Ritchie
 - Space travel game
 - Slow OS vs. fast game
 - Ken writes a small OS kernel (UNIX)
 - Ritchie writes a programming language: C, then UNIX in C code
 - MULTICS -> UNICS





The main branches of the UNIX family



UNIX (\neq Linux) distributions (incomplete list)

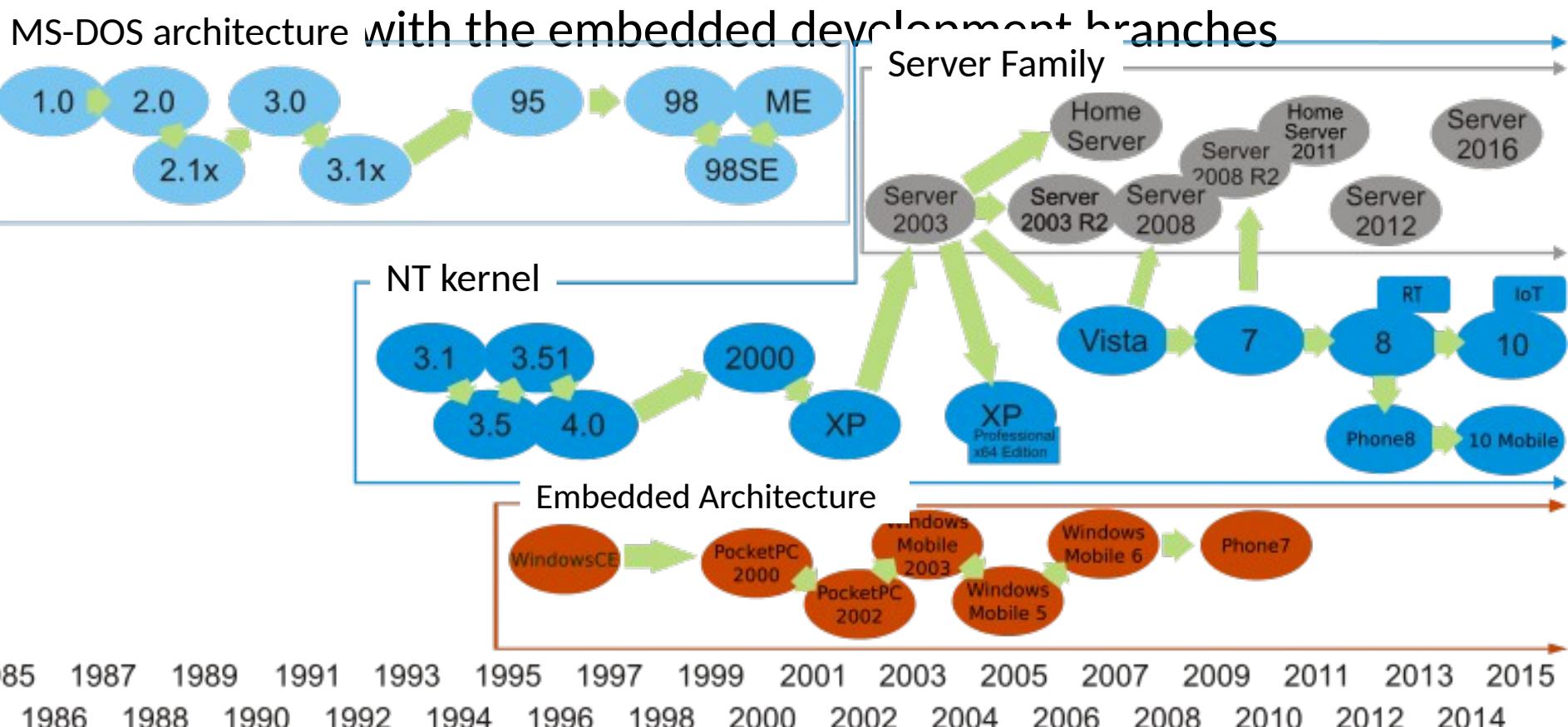
- Up to date catalog: <http://distrowatch.com/>
- On servers
 - RedHat Enterprise Linux and clones (CentOS, Scientific Linux)
 - SUSE Linux Enterprise Server / openSUSE (Novell)
 - Debian
 - Ubuntu Server
 - OpenBSD and *siblings* (FreeBSD, NetBSD, ...)
 - Oracle (Sun) Solaris and other versions (OpenSolaris)
- On clients
 - Ubuntu (Kubuntu, Edubuntu, Mythbuntu, ...)
 - Linux Mint
 - Arch Linux
 - Fedora (RHEL „sandbox”)
 - SUSE Linux Enterprise Desktop / openSUSE (Novell)
- In embedded systems
 - Android-based systems (phones, tablets, TV-s, TV boxes, etc.)
 - Tizen, OpenWrt, Embedded Debian, Arch Linux for ARM, etc.
 - OpenElec.tv (only for further reading)

The personal computer (PC) era - War without control



Evolution of Windows

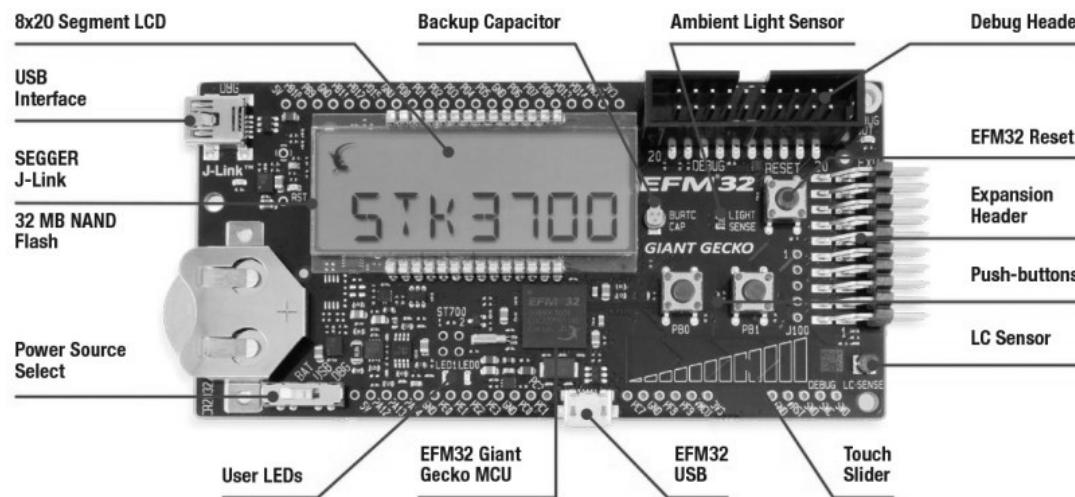
- Graphical User Interface (GUI) for MS-DOS and for the Apple
- Multimedia applications (Win 3.1)
- Refurbished kernel design (Windows NT) <- IBM OS/2, DEC VMS



Embedded OS

- Specific task oriented systems, embedded in the environment
 - Many applications: vehicles, industrial and medical equipments, TV-s, smartphones
 - Typical requirements: real-time, deterministic, and safety critical operation
 - Multiple HW architectures, broad range of performance
- Examples
 - VxWorks (real-time, commercial)
 - Besides the OS, it provides a development and simulation environment

- Mars Pathfinder (1997) (classic example of the errors based on priority inversion)
- FreeRTOS (real-time, open-source, discussed on Lab.)
 - It is able to maintain hard real-time requirements
 - Complex services
- MicroC/OS (real-time, open-source)
 - Available on many platforms, complex services
- RT-Linux (real-time, open-source)
 - Real-time kernel extension for Linux (two kernels)
 - Applications are running in kernel mode
- Etc
 - VRTX, eCOS, QNX, Windows CE, ...



Classification of the operating systems

- Based on the user tasks' nature and purpose
 - Client: GUI-s and client programs are emphasized
 - Server: providing server features (web, mail, database, etc.)
 - Embedded: making smarter industrial or household equipments
 - Others: GRID systems, Storage systems
- Based on the OS (and kernel) properties
 - User interface: GUI or terminal
 - Kernel structure: monolithic, microkernel, exokernel, multikernel (will be discussed later)
 - Kernel operation: real-time, non real-time
 - CPU support: x86, ARM, etc, and multi-architecture, SMP, multiprocessor (100+)
 - License: closed or open-source (GPL, BSD licenses)
 - Communication: network and distributed (supports and provides such services)
- Def.: real-time system
 - The response time to a query (specific) will be less than a given time period
 - Hard real-time: with the probability of 1
 - (soft) real-time: with high probability

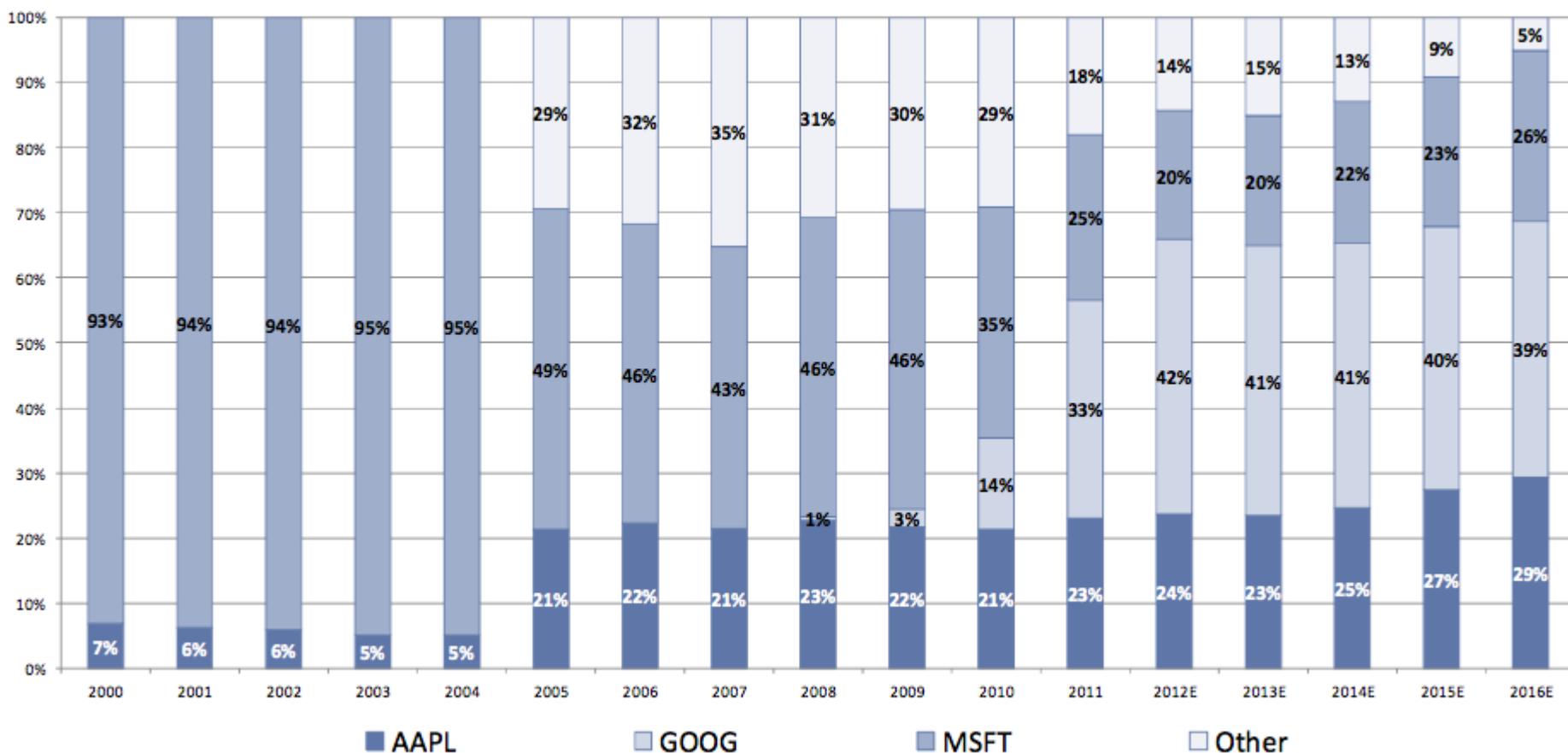
Market share of the OS-s

- On clients
 - The smartphones and tablets are changing the OS market dramatically
 - PC: Windows 92%, Mac OSX 6,4%, Linux 1,6%
(few years ago: Windows 94%, Max 5%, Linux 1%)
 - PC+phones+tablets: Windows 20%, OSX/iOS (UNIX) 24%, Android (UNIX) 42%
 - See next slide!
 - Up to date charts:
 - http://en.wikipedia.org/wiki/Usage_share_of_operating_systems
- On servers
 - Unix 68%, Windows 32%

Market share of the OS-s

Exhibit 1: Vendor share of consumer compute, 2000-2016E

Shift from single-vendor dominance (MSFT) to multiple vendors (AAPL, GOOG, MSFT, Other)

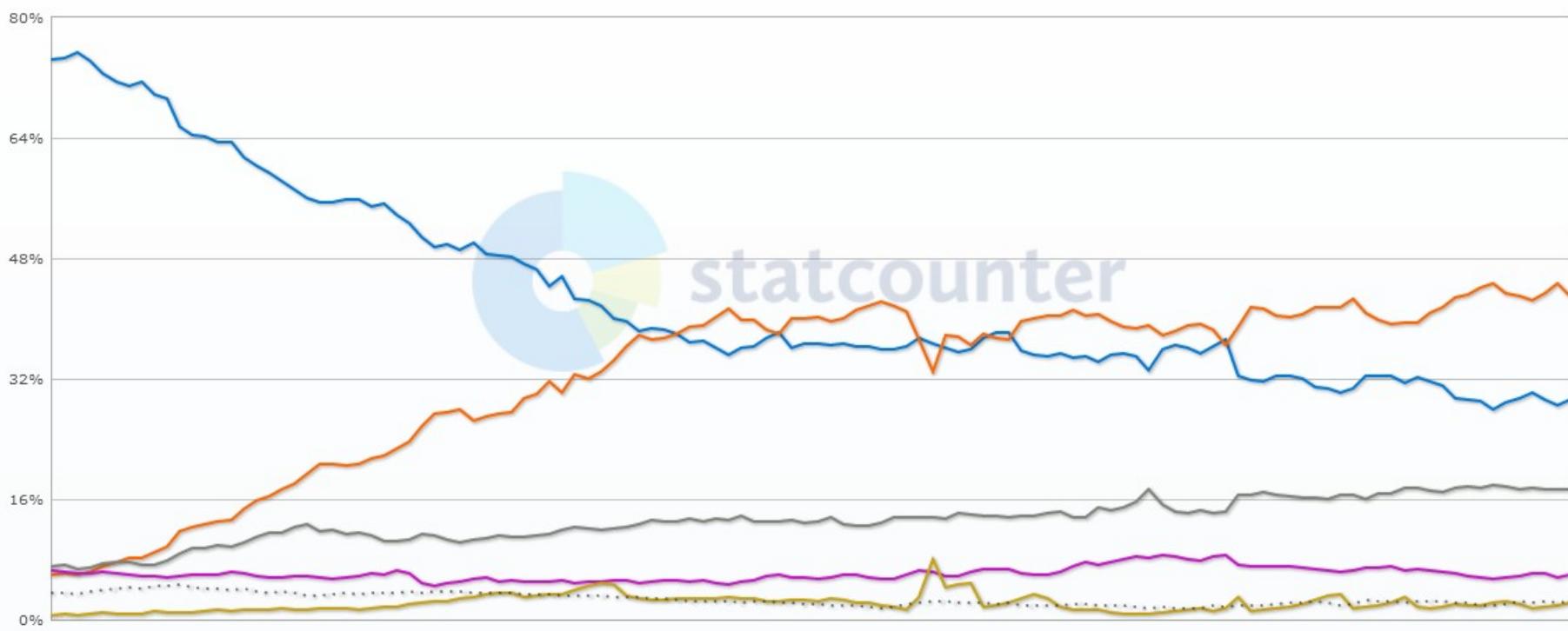


Source: IDC, Goldman Sachs Research.

Market share of the OS-s

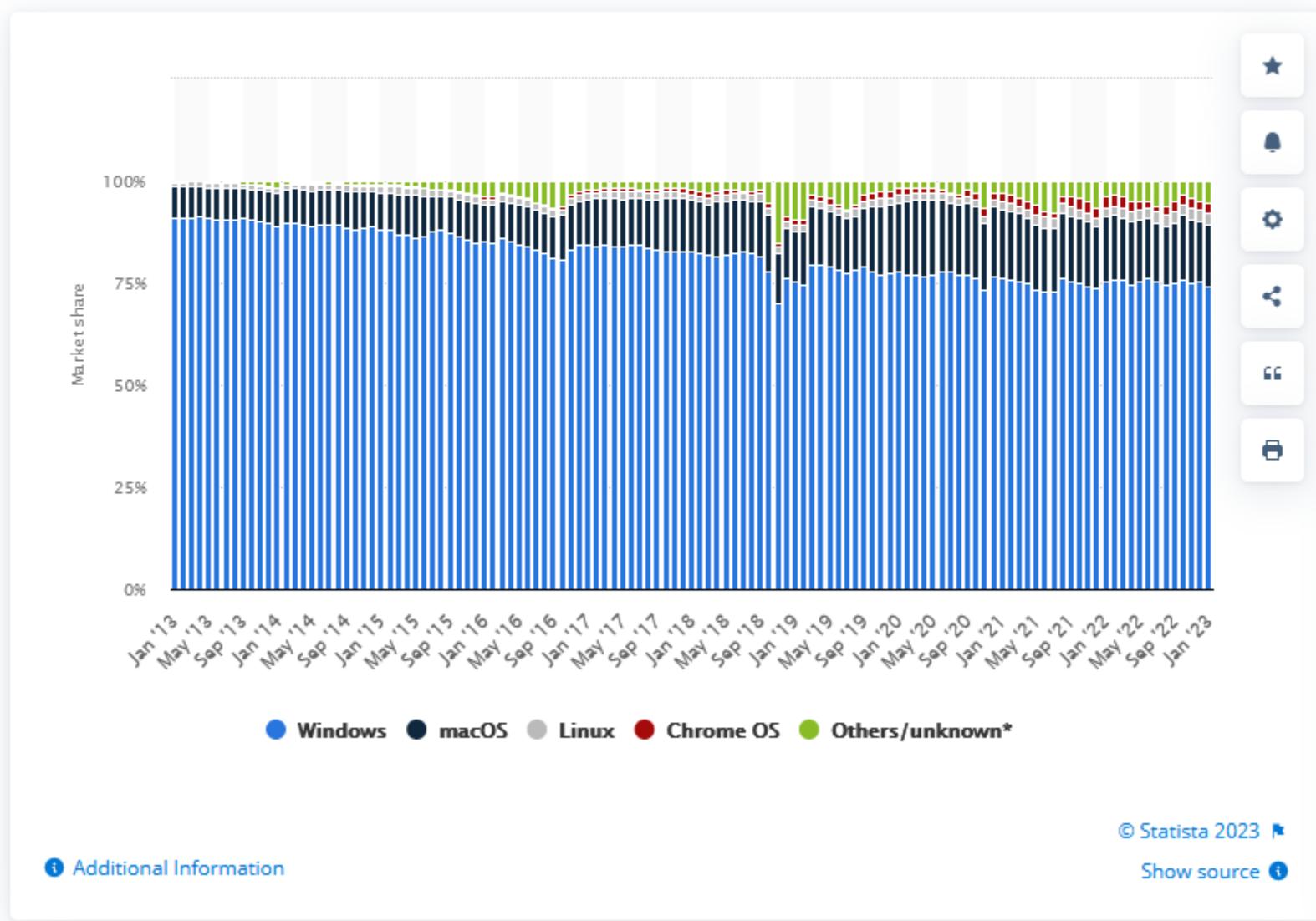
Operating System Market Share Worldwide
Feb 2013 - Jan 2023

Edit Chart Data



Global market share held by operating systems for desktop P

January 2023



Market share of the OS-s

Global Server Operating System Market Share, By Operating System, 2021



www.fortunebusinessinsights.com

What makes order in the chaos of OS-s?

- Wide range of HW-s and SW-s
 - The nightmare of application developers
- The open-systems approach
 - Interoperability
 - Portability
 - User „portability”
- Implementing open-systems: standardization
 - Standard (de jure): specified by a formal association and approved in a formal way
 - Standardization of interfaces and protocols
 - Testing of conformity: checking implementations and applications
- Associations: ISO/IEC, IEEE, W3C, ANSI, Open Group, etc.
- OS standards (examples)
 - IEEE POSIX standard family
 - Linux Standards Base ([LSB](#))
 - AT&T SVID (pl. SVR4) and BSD
 - Open Group: X/Open, Unix95, Unix98, ...

Trends

- Better and cheaper HW-s
 - More processor cores, less power consumption
 - Heterogeneous architectures (multi type cores, OpenCL, FPGA threads)
 - Changing throughputs (SSD, fast flash IC-s)
- OS development trends
 - Multicore kernels (**multikernel**)
 - distributed system of **OSNodes**
 - Reducing the kernel overhead (**exokernel**)
 - the kernel makes a longer data route between the processor and devices
 - minimizing the kernel tasks, and some parts of the kernel runs on user level
 - The kernels are usually built around the processor
 - **universal memory**: the fast flash disks can become primary memories
 - CPU address base instead of [device ID + block] addressing
 - The memory is tied to multiple processor cores
 - The architecture is more and more built around the memory
 - New type of file systems
 - Extremely distributed – e.g. Google, object repository

Summary

- The OS is complex ecosystem
 - It's purpose to manage HW devices and perform user tasks
 - Also provides many additional services
 - Wide range of usages on wide range of HWs
- The basic ideas are simple
 - Purpose oriented, secure, compatible, cooperative, open, easily programmable
- Colorful systems
 - From a few thousand LoC to multi-million LoC systems
 - Different services and user interfaces
 - The standardization is important (open interfaces and protocols)
- During this semester
 - An introduction will be provided to the different details of this ecosystem
 - Further knowledge can be obtained by trying some of the tasks in a virtualized environment (without risks)