ICT WLW

FHNW Fachhochschule Nordwestschweiz

Workshop Linux und Webtechnologien
→ 1. Teil Linux

Matthias Meier

Ziele (gem. Lehrplan)

- Lernziel, Kompetenzen
  - *Einführung in das Betriebssystem LINUX anhand von praxisnahen Versuchen.*
  - *Beherrschen der gängigen Befehle und Werkzeuge sowie Befähigung zur Installation und Administration von Anwendungen*
- Lernmethoden
  - *Selbständiges Praktikum mit Begleitung*
    - *Sowohl "**Ubuntu Desktop**" auf Ihrem Notebook*
      - *virtualisiert, d.h. als «Gastsystem»*
    - *Als auch "**headless**" auf einem **Raspberry Pi***
- Leistungtest
  - *Multiple Choice Kurztests*
    - *Ohne Unterlagen!*
    - *Ab. 2. Unterrichtsblock jeweils zu Unterrichtsbeginn*
    - *Es sind alle Kurztests unter Anwesenheit des Dozenten durchzuführen!*

Lerninhalt

- Einführung in GNU/Linux
  - *Geschichtliches, Rechtliches*
  - *Grafische Oberfläche (GUI), X-Windows, Anwendungen, ...*
  - *Zusammenhänge Filesystem, Boot-Vorgang, Modules, ...*
  - *Gängige Unix-Commands*
- Systemadministration
  - *Installation / User-Administration / Filesystem-Administration*
  - *Applikationen installieren / kompilieren*
  - *Linux als Server (LAN und WAN/Internet)*
  - *Remote-Access*
- Script Programmierung
  - *ein wenig Shell-Scripting*
  - *sowie Python Scripting*

UNIX – Geschichte

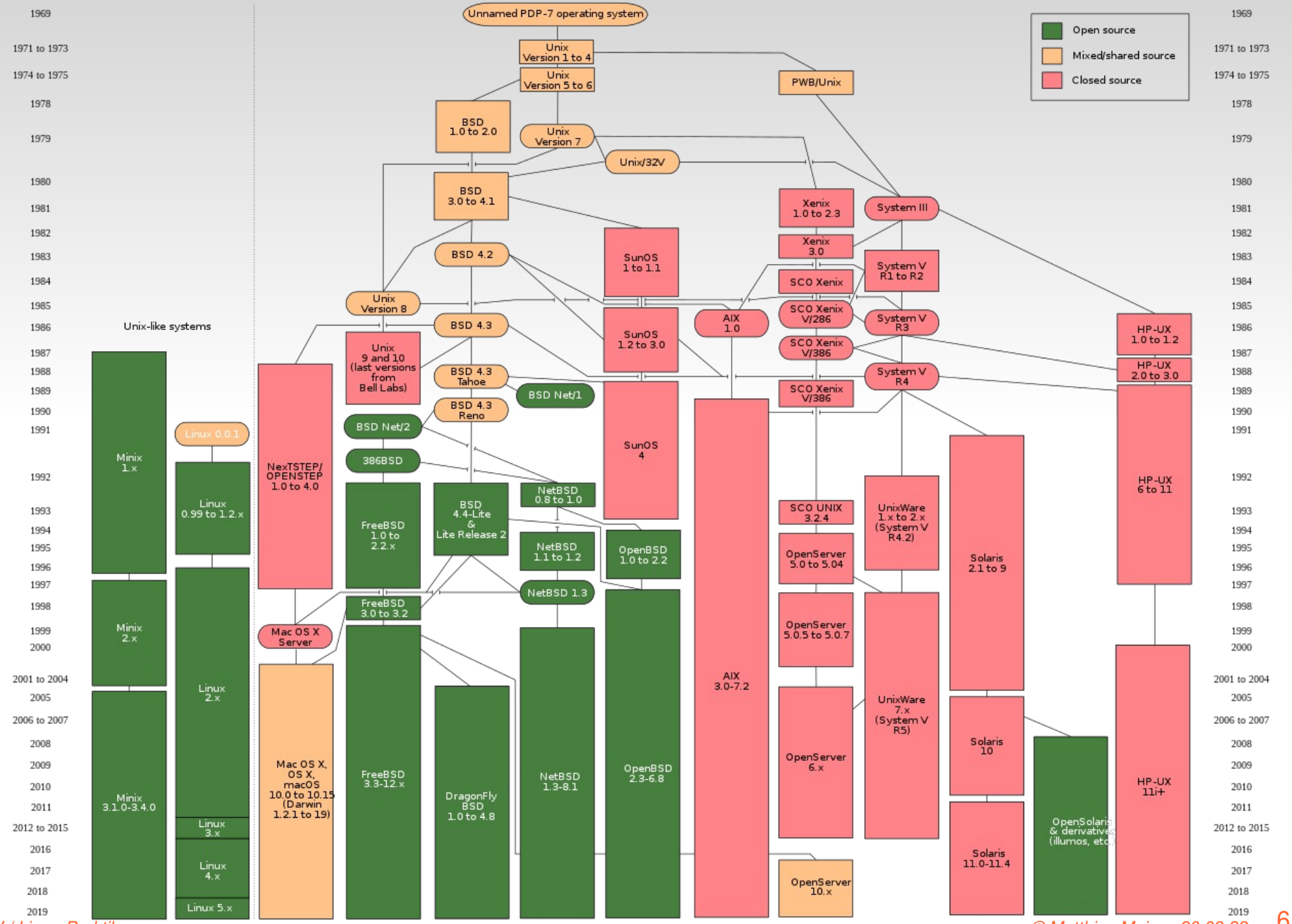
http://de.wikipedia.org/wiki/Geschichte_von_Unix

- 1969: erstes „UNICS“, in den Bell Labs (später AT&T)
  - *Noch in Assembler, auf einem ausgedientem PDP7 http://de.wikipedia.org/wiki/PDP-7*
  - *UNIX-Väter: Ken Thompson und Dennis Ritchie*
- 1972: UNIX wird „portabel“
  - *Portierung auf ein neues leistungsfühigeres System: PDP11 https://de.wikipedia.org/wiki/PDP-11*
  - *Dennis Ritchie entwarf zwecks einfacherer Portierung die Sprache C*
  - *Unterstützte nun: Multiuser, Multitasking und Netzwerk → „UNIX Time Sharing System“*
- 1975: UNIX-Sourcode wird für Universitäten freigegeben:
  - *Universität Berkeley entwickelte basierend darauf das „Berkeley-UNIX“*
  - *Die verwendete AT&T-Quellen waren aber nicht freie Open-Source (somit nur für UNIs zulässig)*
  - *weshalb später in Berkeley-UNIX alle original UNIX-Sources wieder entfernt und ersetzt wurden:*
  - *dieser „freie“ (Open Source) UNIX-Clone erscheint 1977 als „**Berkeley Software Distribution**“ (**BSD**)*
  - *worauf die Workstation Firma SUN später z.B. das **Sun OS** aufbaut...*
- ab 1979: AT&T vermarktet UNIX-Sourcode kommerziell:
  - *Aufgrund der relativ günstige AT&T-Lizenzierung des Sourcecodes entstehen viele kostenpflichtige UNIX-Derivate: XENIX, SINIX, SUN Solaris, HP-UX, AIX, SCO*
- *1988: AT&T „UNIX System V Release 4“*
  - *AT&T versucht die Wiedervereinigung der verschiedenen AT&T-UNIX-Abkömmlinge*
- *ab ca. 1990 werden die POSIX-Standards definiert*
  - *Definieren Standards für das UNIX Programmier-API , Tools und Libraries (kein Sourcecode!)*

Grobübersicht Entwicklung / Open- vs. Closed Source

Quelle: Von Eraserhead1, Infinity0, Sav_vas - Levenez Unix History Diagram, Information on the history of IBM's AIX on ibm.com, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=1801948

Deutlich detaillierter → *http://www.levenez.com/unix/unix.pdf*



LINUX (LINus UniX)

Siehe http://en.wikipedia.org/wiki/History_of_Linux u. Biographie Linus Torvalds Biography: „Just For Fun“

- 1991, begonnen als Hobby des damals finnischen Studenten **Linus Torvalds**
  - *Ursprünglich Idee: ein Terminal-Emulations-Programm zwecks Zugriff auf den UNI-Campus*
  - *Bald entstand aber die Idee, einen Clone des UNIX-Kernels für PCs zu entwickeln*
  - *Linus zog schon bald die Internet-Community in die Entwicklung mit ein*
    (ursprünglich über die Newsgroup von MINIX...)
- **Funktional** angelehnt an **UNIX** (welches nicht auf PCs lauffähig war) und **MINIX**
  - *MINIX war ein einfaches UNIX-kompatibles Lehrbetriebssystem für Mikrocomputer vom Datacom- und OS Professor Andrew S. Tanenbaum*
    - *unvollständig und nur zu Ausbildungszwecke!*
  - *MINIX war keine „freie Opensource“ (und auch nicht auf AT&T-Quellen basierend)*
- Linus zielte mit seinem OS hingegen auf ein **Community-Projekt** mit **POSIX-Konformität**
  - *Quellcode-Kompatibilität, sodass UNIX-Anwendungen für Linux bloss neu übersetzt werden mussten*
  - *Damit war von vornherein viel Software für Linux verfügbar*
    - *GNU-Compiler (GCC), Bash Shell, div. Commandline Tools aus den GNU- und BSD-Projekten*
- Linus entschied sich für die **GPL-Lizenz** (**G**NU **P**ublic **L**icense):
  - *GPL garantierte lizenzrechtlich eine „Evolution“ des Produktes*
  - *durch lizenzrechtlich festgelegten freien Zugang zum Sourcecode bei Weitergabe des Binärcodes...*
- **Linus Torvalds** behält immer noch (!) **Oberaufsicht über Linux**...
  - *mit Unterstützung einer riesigen der Open Source Entwicklergemeinde, mittlerweile vorwiegend namhafte kommerzielle Firmen!*

Linux - Who is doing the Work Jahre 2015 .. 2018

Quelle: https://www.linuxfoundation.org/events/2016/08/linux-kernel-development-2016/

The most active companies over the 3.19 to 4.7 development cycles were:

Company	Changes	Percent
Intel	14,384	12.9%
Red Hat	8,987	8.0%
none	8,571	7.7%
unknown	7,582	6.8%
Linaro	4,515	4.0%
Samsung	4,338	3.9%
SUSE	3,619	3.2%
IBM	2,995	2.7%
consultants	2,938	2.6%
Renesas Electronics	2,239	2.0%
Google	2,203	2.0%
AMD	2,100	1.9%
Texas Instruments	1,917	1.7%
ARM	1,617	1.4%
Oracle	1,528	1.4%

Company	Changes	Percent
Outreachy	1,524	1.4%
Vision Engraving Systems	1,456	1.3%
Free Electrons	1,453	1.3%
NXP Semiconductors	1,445	1.3%
Mellanox	1,404	1.3%
Atmel	1,362	1.2%
Broadcom	1,237	1.1%
NVidia	1,146	1.0%
Code Aurora Forum	1,033	0.9%
Imagination Technologies	963	0.9%
Huawei Technologies	937	0.8%
Facebook	877	0.8%
Pengutronix	790	0.7%
Cisco	692	0.6%
Qualcomm	656	0.6%

Vgl. Datei MAINTAINERS im Linux Source Tree: **grep** "M:" MAINTAINERS | **sort** -u | **wc** -l
--> ca. 1500 „Maintainers“ von Linux-Subsytemen und ca. 15'000 Entwickler insgesamt

GNU - www.gnu.org

- **Richard Stallman**
  - *hatte lange vor Linus die Vision einer **Open Source Community***
- initialisierte 1980 das **GNU-Projekt** **(also 11J vor LINUX)**
  - ***GNU = G**nu is **N**ot **U**nix* *(eine Wortspielerei)*
  - *mit dem Ziel ein **freier UNIX-Clone** inkl. Compiler und Dienstprogrammen durch die „Open Comunity“ zu erstellen (also nicht wie Linux nur ein Kernel)*
  - *Nicht einfach just for fun sondern aus einer Vision dass Software frei sein sollte...*
- Gründete dazu die ***Free Software Foundation*** *(www.fsf.org)*
  - *Die FSF unterstützte u.a. die rechtlichen Aspekte von „Open Source“*
  - *(die FSF nicht mit der „Open Software Foundation“ (nun „The Open Group“) verwechseln mit Ziel „making Standards Work“...)*
- Free im Sinn von von **Freiheit!** (nicht bloss/zwingend Gratis)
  - *Freiheit bezüglich Programmausführung,*
  - *Freiheit bez. Veränderbarkeit,*
  - *Frei Kopien weiterzureichen (gratis oder gegen Gebühr),*
  - *und Freiheit, veränderte Versionen weiterzugeben*
- Bedeutende GNU-Projekt sind:
  - *der GNU C-Compiler (**GCC**), die C Standard Libraries (**glibc**) und viele Dienstprogramme (**GNU-Tools**) sowie die GNU Lizenzformen (**GPL**, LGPL)*
  - *jedoch nicht (!) der GNU Kernel „Hurd“ welcher keine Bedeutung hat*

Fazit GNU/Linux

- **Linux** ist also „nur“ ein (UNIX-kompatibler) Betriebssystem-Kern

- LINUX-Systeme verwenden zudem üblicherweise...
  - *viele **GNU- und BSD-Tools** (Shell, Compiler, UNIX-komp. Commands)*
  - *oft eine graphische Oberfläche basierend auf andere Open Source Projekte (X-Server (X.Org, Wayland), Desktop Environment (GNOME, KDE, ...) etc.*
  - *Open Source macht's möglich!*

  - *Richard Stallmann wünschte deshalb die Bezeichnung: „**GNU/Linux**“*
  - *Eigentlich müsste die System-Bezeichnung jedoch z.B. lauten: „**GNOME/X.Org/BSD/GNU/APACHE/.../.../.../.../.../.../.../.../Linux**“*

LINUX-Distributionen

- LINUX ist nur ein Betriebssystem-**Kern** (Kernel)
  - *Daneben werden Dienstprogramme, grafische Oberfläche etc. benötigt*
  - *Die Linux-**Distributionen** bündeln den übersetzten Linux-Kernel mit anderen übersetzten (in der Regel Open Source-) Programmen*
  - ***Damit wurde Linux erst für „Normalanwender“ anwendbar***

- Einige Linux-Distributionen (vgl. http://distrowatch.com/)
- Für Einsteiger geeignet:
  - ***Ubuntu** (**und deren Schwesterprojekte Kubuntu, Lubuntu, Xubuntu, ...) - basiert auf 'Debian', gesponsert von Fa. Canonicals***
  - ***Mint** – basiert wiederum auf Ubuntu (Ubuntu hat Zeit die „ältere“ Linux-Gemeinde etwas verärgert…)*
  - ***Xubuntu, Lubuntu** – für schwache/ältere PCs&Notebooks*
- Für Erfahrene:
  - ***Debian** – v.a. auch im Serverbereich vertreten da sehr seriös, lange Releasezyklen (2J+), viele Platformen, „clean OSS“*
  - ***Fedora** und **Cent OS***
  - ***Arch Linux** (Sehr nahe am Quellcode, für „Enthusiasten“) und **Manjaro** (ein einfacher installierbares Arch Linux Derivat)*
- Kommerzielle Distributionen – kostenpflichtig, für Firmen geeignet, wenn langfristiger Support nötig:
  - ***RedHat Linux** (von Redhat)*
  - ***SuSE Enterprise** (von Oracle)*
- **Spezialisierte Distributionen:**
  - ***System Rescue** u.**Virenscan** Images (zur Systemreperatur, HDs clonen, ...) wie z.B. c't **Desinfec't***
  - *Distributionen für "**Ethical Hacking**" wie z.B. **Kali Linux***

Verbreitung von Linux

(s.a. http://en.wikipedia.org/wiki/Usage_share_of_operating_systems)

- Verbreitung verschiedener Linux-Distributionen http://distrowatch.com/
  - *Genauigkeit dieser Zahlen jedoch schlecht…*

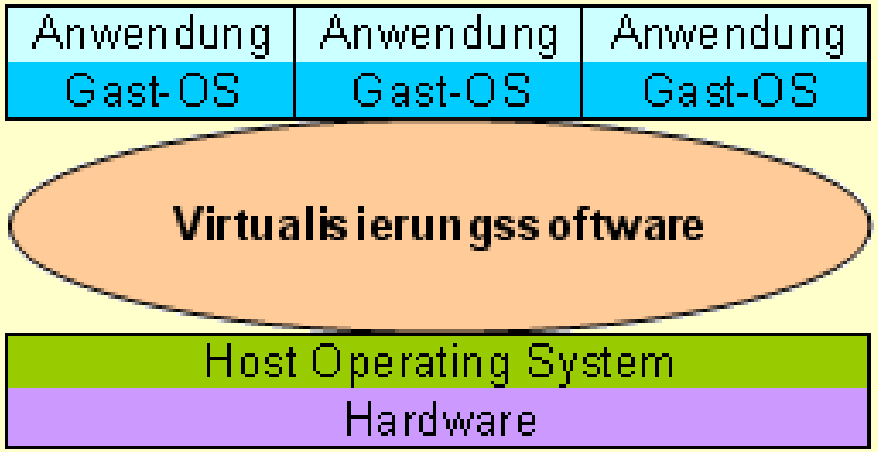
- Im Vergleich zu anderen Betriebssystemen **dominiert Linux insgesamt**...
  - *im Internet-Serverbereich (Web- und Cloudserver, Suchmaschinen, CMS)*
  - *im Mobile-Bereich (Android verwendet Linux als OS-Kernel!)*
  - *im SOHO Bereich (Small and Home Office) für Router und NAS*
  - *im Consumer- und Industriellen Bereich (generell Internetfähige Geräte wie TVs, Navis, VoIP, Internet-Radio, ...)*
  - *sowie im Supercomputer-Bereich*

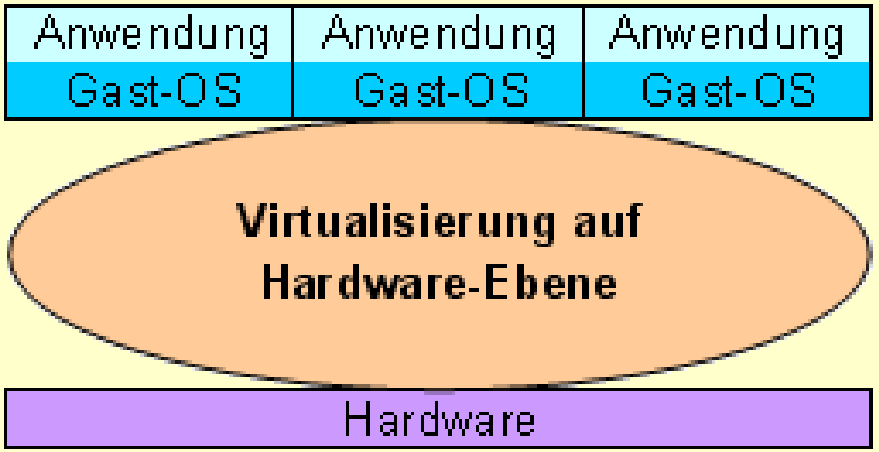
- Linux ist also extrem skalierbar von kleinen bis sehr grossen Systemen!

- Linux ist (nur) auf dem **Desktop-OS-Markt** schwach vertreten:
  - *Linux ~ 2%, OS-X ~10%, Windows ~98%*

Virtualisierungslösungnen
Hypervisor, Virtual Machine Monitor (VMM)

- (Hypervisor Typ 2)
  Virtualisierungssoftware unter Kontrolle einen Host Betriebssystems
  - *einfach einzurichten*
  - *gaukelt dem „Gastsytem“ virtuelle Hardware vor*
  - *diverse Produkte (Virtualbox, Parallels, VMware Player, MS Virtual PC, …)*
  - ***v.a. im Desktopbereich** vertreten*
- (Hypervisor Typ 1)
  Virtualisierung auf Hardware-Ebene
  - *etwas effizienter*
  - *diverse Produkte (XEN, KVM, VMware ESX/ESXi, …)*
  - *managed geordneten HW-Zugriff*
  - ***v.a. im Serverbereich** vertreten*





Bildquelle: http://de.wikipedia.org/wiki/Virtualisierung_%28Informatik%29

Andere UNIX-Clones

- keine der folgenden UNIX-Clones basiert auf UNIX Quellcode (wie auch Linux nicht!)
- UNIX-Clones basierend auf **BSD** (Berkeley Software Distribution):
  - *Viele! Vgl. http://de.wikipedia.org/wiki/Vergleich_von_BSD-Betriebssystemen*
  - *Die BSD-Derivate beinhalten im Gegensatz zum LINUX-Projekt **nicht nur** den **Kernel** sondern auch die **BSD-Dienstprogramme***
    - ***auch** GNU/Linux-Distributionen verwenden ja etliche BSD-Tools*
  - *Die **BSD-Lizenz** ist **weniger streng als** die **GPL***
    - *d.h. man darf legal den BSD Code weiterentwickeln und als „Closed Source“ vertreiben*
    - *SUN durfte deshalb SUN Solaris und Apple das Mac OS-X resp. dessen Unterbau „Darwin“ auf Basis BSD weiterentwickeln/modifizieren **ohne Auflage, den modifizierten Quellcode offenzulegen!***
- **GNU 'Hurd'**
  - *hat sich nie durchgesetzt*
- **MINIX**
  - *hat sich nie durchgesetzt da als Lernsystem konzipiert (funktional eingeschränkt) und lange nicht freie Open Source*

Open Source - Idee

- Jeder kann mit Sourcecode experimentieren...
  - *und z.B. Fehler suchen*
  - *Das Produkt hat damit schnell eine hohe Qualität!*
- Gründer eines Projektes ...
  - *behalten in der Regel Oberaufsicht (=Maintainer)*
  - *entwickeln das Produkt selbst aktiv weiter*
  - *integrieren Bugfixes und neue Features aufgrund von Vorschlägen (nach erfolgter Prüfung)*
- Oder es entstehen neue Projekte oder Derivate
- An der Front wird nur der (vorzugsweise POSIX-konforme) Sourcecode weitergegeben...
  - *Damit ist das Projekt für POSIX-Konforme Unix-Derivate sowie in Grenzen auch für Windows portierbar*
    - *Erfordert geeigneten Compiler (meist GCC)*
  - *Distributionen beinhalten übersetzte Programme (Binary Code)*
- Open Source Hosting z.B. http://sourceforge.net

www.gnu.org

- Die **GPL** (GNU Public Licence)
  - *ist eine Open Source Lizenz mit einem **starken Copyleft***
    - *die Lizenzart und der Lizenztext darf nicht geändert werden!*
    - *Weiterentwicklungen von GNU-lizenziertem Quellcode muss auch wieder unter GPL lizenziert werden!*
  - *Bei **Weitergabe von Binärcode** (kostenlos oder gegen Gebühr) muss dem Binärcode-Empfänger auf verlangen **der gesamte Quellcode** kostenlos ausgeliefert werden!*
  - *Die GPL betrifft **alles was im gleichen Adressraum** läuft*
    - *Der Linux-Kernel und Anwendungen laufen jeweils in anderen virtuellen Adressräumen*
- **LGPL** – Lesser GPL
  - ***schwaches Copyleft***
    - *Die Grenze ist nicht der Adressraum sondern nur das betreffende Software-Projekt*
  - *Gerne verwendet für Libraries*
    - *LGPL-Libraries lassen das Linken von nicht-GPL Programmen zu*
    - *Insbesondere ide GNU Standard-C-Libraries (**glibc**) steht unter LGPL*
- Es existieren aber auch viele Open Source Lizenzen,
  - *insbes. ohne Copyleft wie z.B. **BSD-License, Apache-License, MIT-License, ...***

Beispiel Virtualbox - Lizenzen

- Mehrfach verkauft: Innotek → SUN → Oracle
  - *Virtualbox gibt's als Open Source + unter kommerzieller Lizenz*
  - *Möglich da alle Quellcode-Rechte bei einer Firma (Oracle) liegen*
- Die kostenlose Variante unterliegt ...
  - *Grösstenteils unter der GPL (GNU Public Lincense)*
  - *Nur das "Virtual Box Extension Pack" unterliegt der "Personal Use and Evaluation License" (PUEL)*
    - *Ist aber wie Virtualbox gratis herunterladbar von http://www.virtualbox.org*
    - *Darf jedoch nicht kommerziell verwendet werden!*
      - *Für kommerzielle Anwendung muss das Extension Pack gekauft werden!*
    - *Das "Virtual Box Extension Pack" beinhaltet Zusatzfeatures wie*
      - *USB 2.0 und USB 3.0 Unterstützung, Remote Desktop, verschlüsselte Disk-Images, ...*

Bestandteile eines kompletten Desktop-Betriebsystems

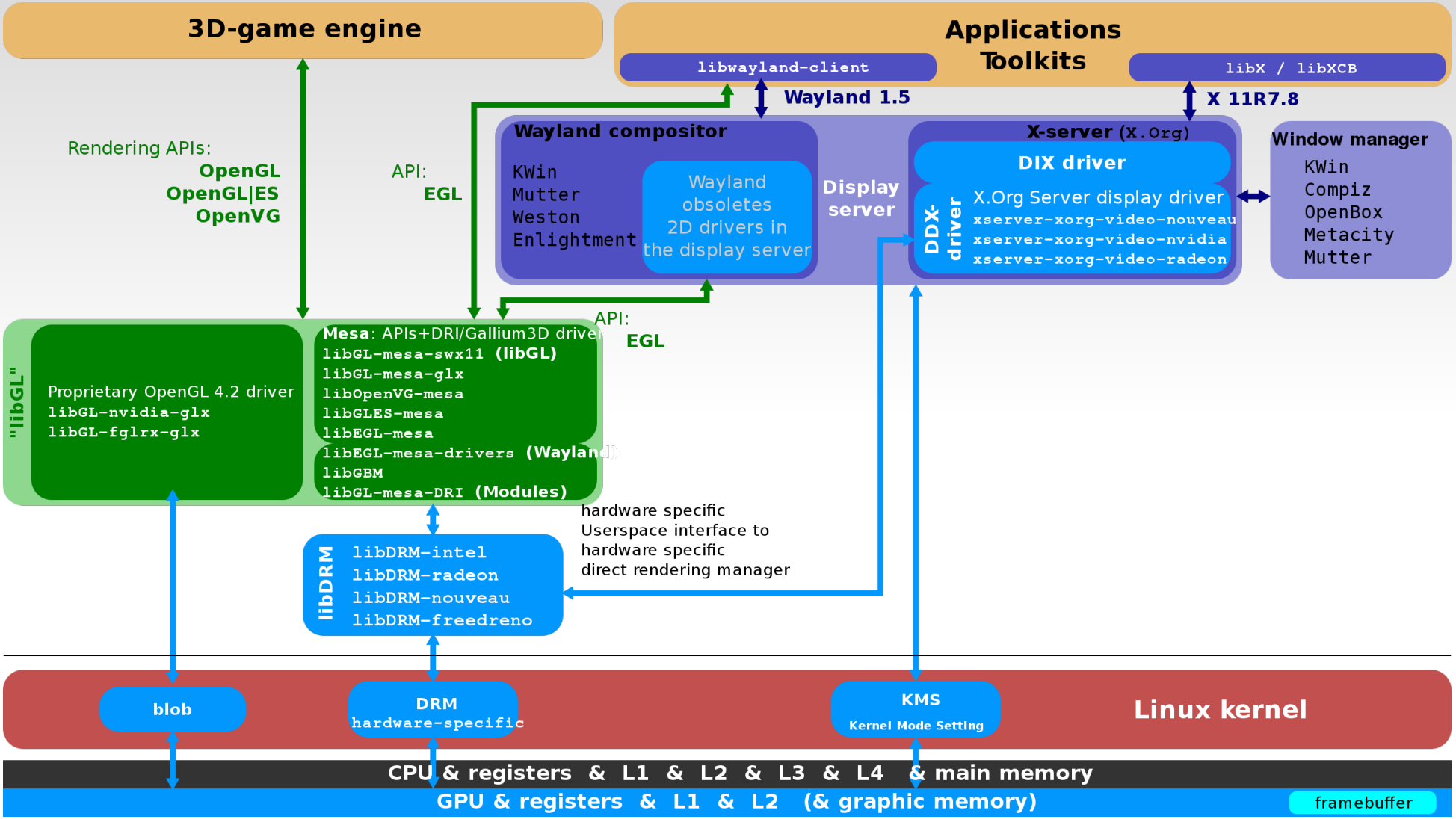
- Ein Betriebsystem-Kern (OS Kernel) z.B. Linux
  - *zwecks Hardware-Abstraktion*
    - *Speicherverwaltung (Paging/Swapping, ...)*
    - *Prozess-Management (Prozesse starten, beenden, Interuprozess-Kommunikation)*
    - *Gerätetreibern (EIDE, SCSI, USB, Tastatur, Maus, MMU,...)*
    - *Filesysteme (EXT2, EXT3, ReiserFS, VFAT, XFS, ...)*
    - *Netzwerk-Funktionalität (Sockets,…)*
    - *Sicherheitsmechanismen*
- viele Dienstprogramme (meist GNU-Tools)
  - *Kommando-Interpreter (Shell wie "sh", "bash", ...)*
    - *Ermöglicht das Laden von Applikationen und ausführen von Shell-Scripts*
  - *Ein Init-System wie z.B. systemd*
  - *viele weitere Dienstprogramme (ls, mkdir, mv, useradd, sudo, …)*
- Graphisches Subsystem (z.B. Ein X-Server wie Xfree86)
    - *X-Window-System ist defacto-Standard auf UNIX*
    - *Xfree86 resp. Xorg sind Open Source Implementation davon*
    - *Graphische Toolkit Librariies (GTK+, QT-Libraries, ...)*
- (Graphisches) „Desktop Environment“ (KDE, Gnome)
    - *Fensterfunktionen (Menus, Taskbar, ...)*
    - *Hilfs- und Dienstprogramme der graphischen Oberfläche*

POSIX-Standards
(Portable Operating System Interface)

- **Durch die POSIX-Standards wird die Portabilität von Applikationen auf Ebene Source Code garantiert**
- **IEEE-Standards 1003.1*** (resp DIN/EN/ISO/IEC 9945)
  - *der Standardisierungsgremien „IEEE“ (Institute of Electrical and Electronics Engineers) und „The Open Group“ (s. www.unix.org , ein Zusammenschluss der früheren X/Open und Free Software Foundation)*
- Standardisiert die **UNIX-Programmierschnittstelle**
  - *also das API (Application Programming Interface) für UNIX (-kompatible) Systeme, insbesondere*
  - *Interface zu System-Bibliotheken (libc, posix-threads,...)*
  - *Interface zum Kernel also die „Systemcalls“*
  - *Funktionsumfang der Shell (Kommandozeilen-Interpreter)*
  - *Verhalten diverser Systemcommands (sh, awk, vi, echo,...)*
  - *http://de.wikipedia.org/wiki/POSIX*
- IEEE-Standards sind leider kostenpflichtig...

Linux Graphics Stack (Overview)

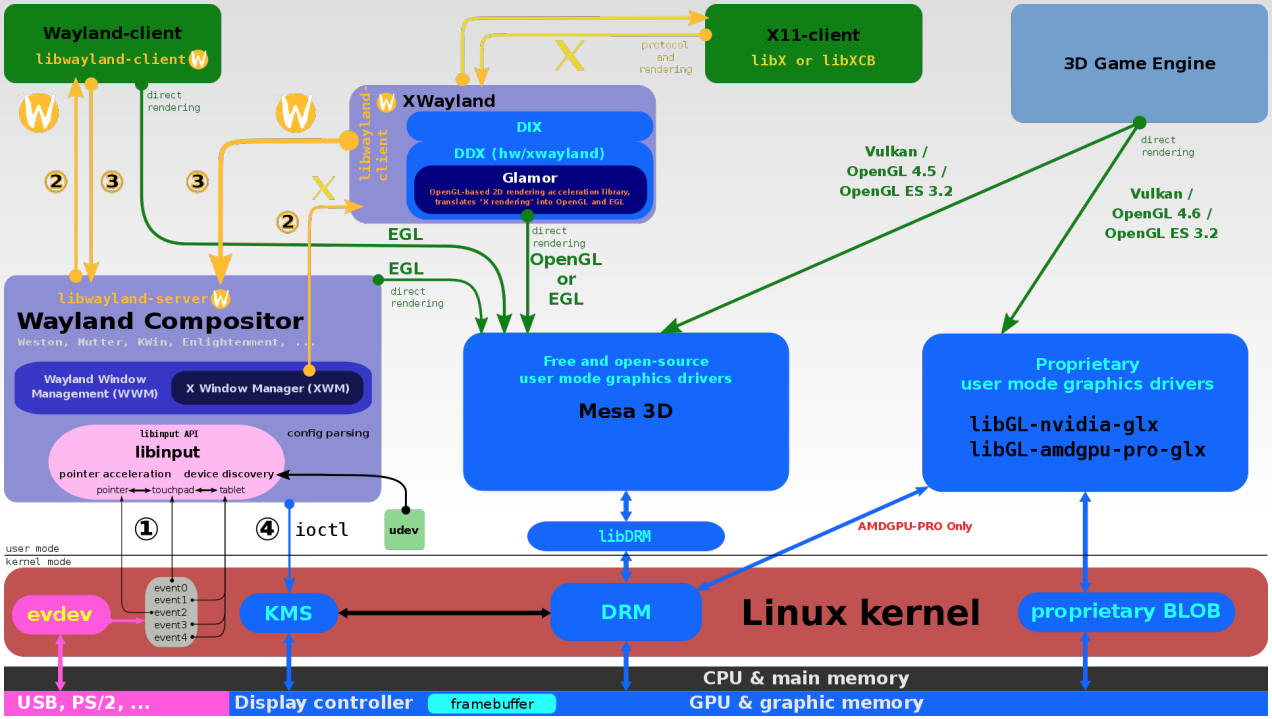
Quelle: https://en.wikipedia.org/wiki/Mesa_(computer_graphics),
https://en.wikipedia.org/wiki/Mode_setting

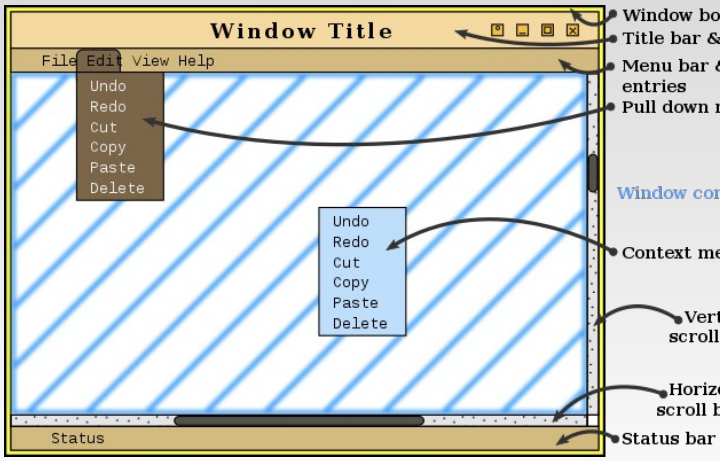


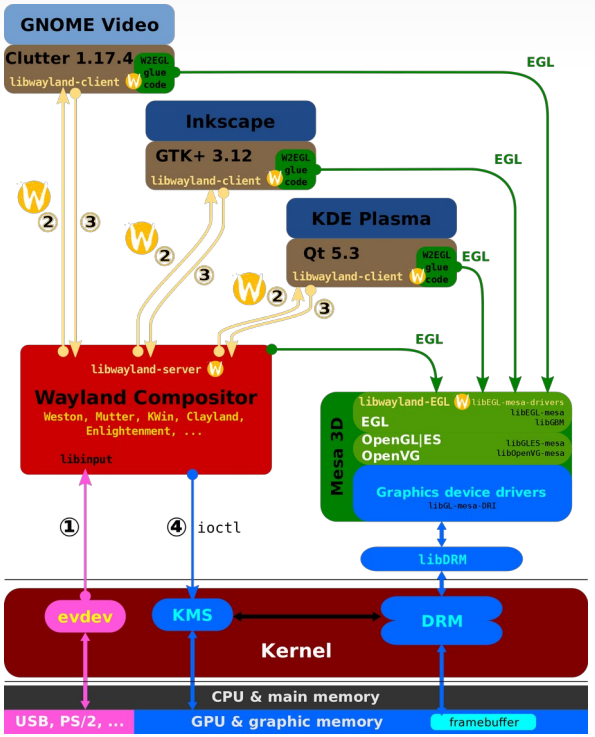
- X11 has 2D drivers in the X11-display server
- Wayland compositors don't (only require access to KMS, OpenGL ES and EGL)

Wayland

vgl. https://en.wikipedia.org/wiki/Wayland_(display_server_protocol)







1)The evdev module of the Linux kernel gets an event and sends it to the Wayland compositor.

2) The Wayland compositor looks through its scenegraph to determine which window should receive the event. The scenegraph corresponds to what is on screen and the Wayland compositor understands the transformations that it may have applied to the elements in the scenegraph. Thus, the Wayland compositor can pick the right window and transform the screen coordinates to window local coordinates, by applying the inverse transformations. The types of transformation that can be applied to a window is only restricted to what the compositor can do, as long as it can compute the inverse transformation for the input events.

3) As in the X case, when the client receives the event, it updates the UI in response. But in the Wayland case, the rendering happens by the client via EGL, and the client just sends a request to the compositor to indicate the region that was updated.

4) The Wayland compositor collects damage requests from its clients and then re-composites the screen. The compositor can then directly issue an ioctl to schedule a pageflip with KMS.

Packages installieren

- 1. Prio **Package-Manager** der Distribution
  - *apt-get, aptitude, Synaptic (auf der graphischen Oberfläche), ...*
  - *Greifen alle auf die gleiche dpkg-Database des Systems!*
  - *Installierte Software ist damit „inventarisiert“*
- 2. Prio: **Zusätzliche Quellen** (für Package Manager)
  - *in /etc/apt/sources.list weitere Quellen aktivieren/zufügen*
    - *„universe“ und „multiverse“*
    - *ev. backports (des nächsten Releases)*
    - *ev. weitere „Fremde Server“*
- 3. Prio: Manuell ***.deb** Package herunterladen/installieren
  - *Vom Internet heruntergeladen – aber nur für installierte Distribution geeignete!*
- 4. Prio: **Closed Source Packages** installieren
  - *Nicht *.deb Package von Nicht-Opensource-Anbietern*
  - *z.B. Jbuilder, Matlab, Poseidon UML,... (auf /usr/local/... oder /opt/...)*
- 5. Prio: **Source-Package aus Distribution ändern**
  - *Source-Package ändern / nneu builden / installieren (auf /usr/local/...)*
- Erst mit letzte Prio und nur wenn nicht anders möglich!
  - *Ab Source-Tarball selber kompilieren und installieren*

Quellcode übersetzen

Zur Kompilation (von Sourcecode) benötigte Packages:

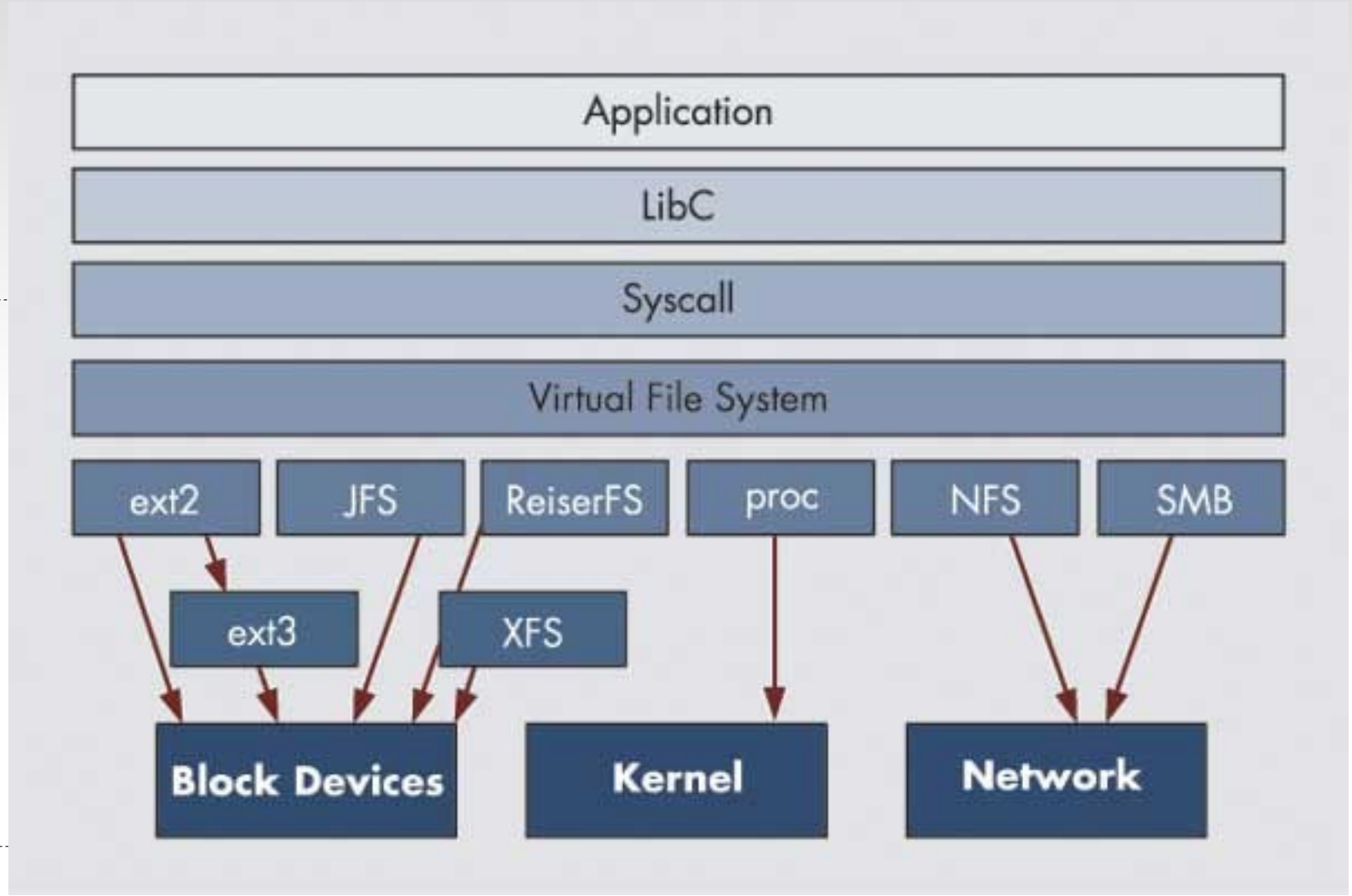
- Compiler und Compiler Tools
  - *Package build-essential*
    - *beinhaltet make, g++ (C++ Compiler), ...*
- Headerfiles den zu verwendeten Libraries
  - *Packages lib****-dev***
  - *Nur anhand der in den Headerfiles enthaltenen Funktionsprototypen kann der C-Compiler ja überprüfen, ob die Funktionsaufrufe in eine verwendete Bibliotheks-Funktion im zu übersetzenden Programm korrekt erfolgt! (korrekter Funktionsname u. Parameterliste)*
  - *Bsp: Die Headerfiles zur Standard C-Library (Pkg. libc6) sind in libc6****-dev***
    - *libc6-dev installiert u.a. das Headerfile /**usr/include/stdio.h** in welchem z.B. für die Funktion **printf()** folgender Funktionsprototyp deklariert wird:*
      *extern int fprintf (FILE *__restrict __stream, __const char *__restrict __format, ...);*
  - *Anmerkung: die Headerfiles werden unter /**usr/include**/, die Binär-Bibliotheken selbst hingegen unter /**lib**/ und /**usr/lib**/ installiert!*
- ev. zusätzliche Tools
  - *z.B. qmake um Programme basierend auf QT-Libraries zu übersetzen*

Einbindung von Filesystemen ins Virtual File System (VFS)

Anwender-Programm

Linux Kernel

Hardware



File System Hierarchy

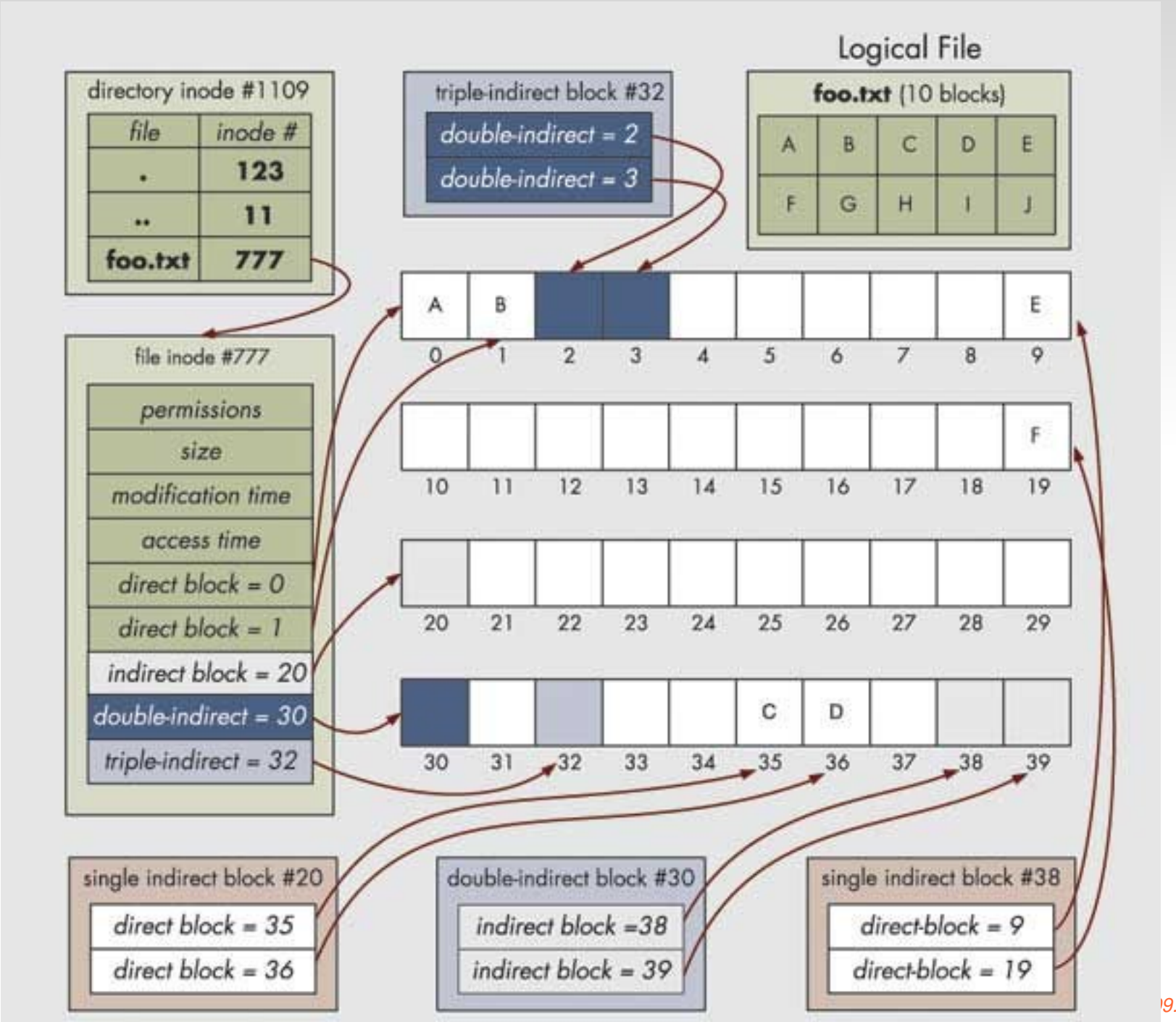
/bin : Essential user command binaries (all users)
/sbin : Essential System binaries
/lib : Essential shared libraries and kernel modules
/boot : Static files of the boot loader
/dev : Device files
/etc : Host-specific system configuration
/home : User home directories
/root : Home directory for the root user
/tmp : Temporary files
/mnt : temp Mount points
/proc : Kernel Interface

/usr : Hierarchy, 2nd major section (sharable, ro)
/var : Hierarchy (Variable Data)
/opt : Add-on application software packages

/usr/bin : Most user commands
/usr/sbin : Non-essential system binaries
/usr/lib : Libraries for programming and pkgs
/usr/share : Architecture-independent data
/usr/local : Local hierarchy
/usr/X11R6 : X Window System, (optional)
/usr/include : standard include files
/usr/src : Source code (optional)

/var/log : Log files and directories
/var/lib : Variable state information
/var/account : Process accounting logs
/var/cache : Application cache data
/var/crash : System crash dumps
/var/games : Variable game data
/var/lock : Lock files
/var/mail : User mailbox files
/var/opt : Variable data for /opt
/var/run : Run-time variable data
/var/spool : Application spool data
/var/tmp : preserved between reboots
/var/yp : NIS database files

Inodes


Logical File
directory inode #1109
file
inode #
.
123
..
11
foo.txt
777
triple-indirect block #32
double-indirect = 2
double-indirect = 3
foo.txt (10 blocks)
A
B
C
D
E
F
G
H
I
J
file inode #777
permissions
size
modification time
access time
direct block = 0
direct block = 1
indirect block = 20
double-indirect = 30
triple-indirect = 32
0 1 2 3 4 5 6 7 8 9
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
single indirect block #20
direct block = 35
direct block = 36
double-indirect block #30
indirect block =38
indirect block = 39
single indirect block #38
direct-block = 9
direct-block = 19

Linux Kernel architecture