Betriebssysteme

it-Akademie Bayern
z/OS und OS/390 Lehrgang 2008

Prof. Dr.-Ing. Wilhelm G. Spruth

Teil 4

Überwacher - Kernel

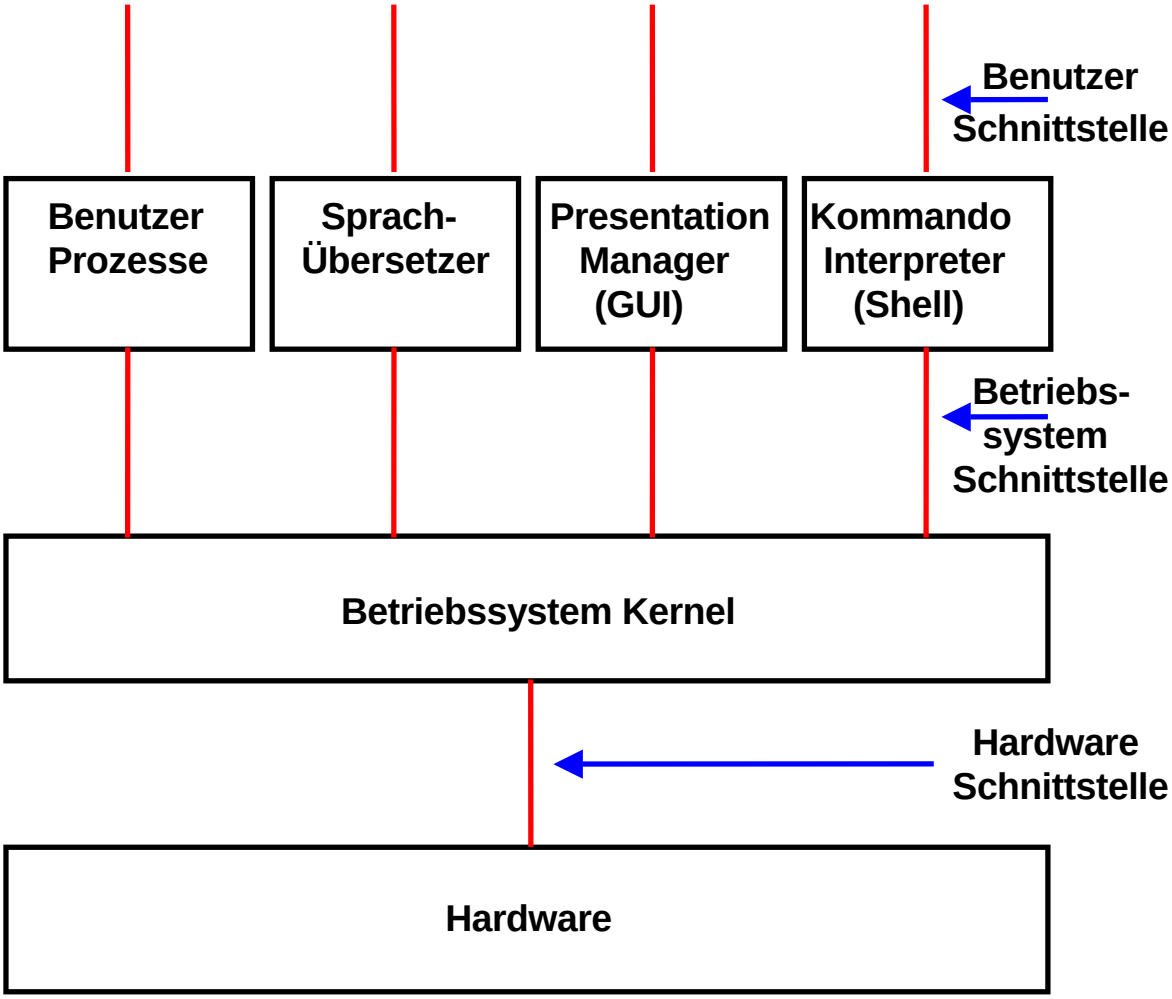
Aufgaben des Betriebssystems

Dem Benutzer wird eine Schnittstelle (Architektur) zur Verfügung gestellt, die leichter zu benutzen und zu programmieren ist als die eigentliche Hardware.

Betriebsmittel (Resourcen) wie

- **Hauptspeicherplatz**
- **Plattenspeicherplatz**
- **Zugriff auf Ein-/Ausgabegeräte**
- **Zeit der CPU**

werden für eine größere Anzahl von Benutzern verwaltet.



Schichtenmodell der Rechnerarchitektur

Der OS/(390 bzw. z/OS Kernel wird auch als „Basic Control Program“ oder BCP bezeichnet.

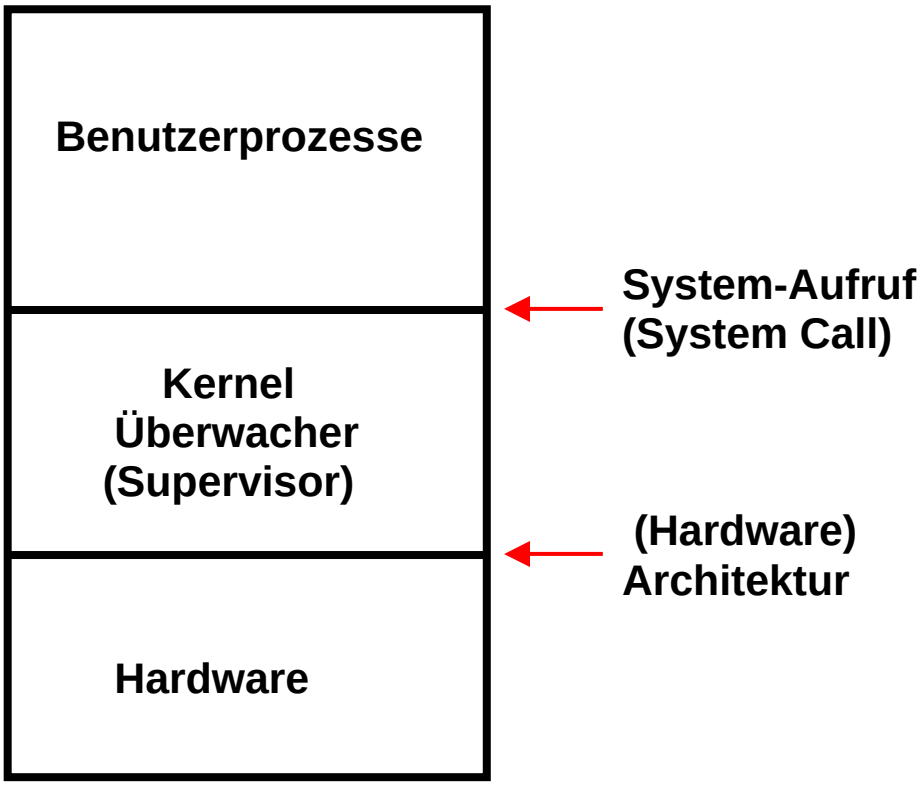
Betriebssystem = Überwacher + Systemprogramme

Das Betriebssystem besteht aus einer zentralen Steuerungs- und Verwaltungsfunktion, dem Überwacher (Supervisor, Kernel), und weiteren Systemprogrammen oder Komponenten, die als „Subsysteme“ bezeichnet werden.

Große Unterschiede bei den Betriebssystemen, welche Funktionen wohin gehören.

Unter z/OS sind diese weiteren Komponenten als Systemprozesse implementiert. Andere Betriebssysteme (z,B. Windows, VAX) kennen hierfür die Funktion des Executive.

Mit Hilfe der Speicherschutzschlüssel des Speicherschutz-Schnellspeichers (Protection Key Store) werden die Systemprozesse vor einem unauthorisierten Zugriff geschützt



Die Hardware reagiert auf die Eingabe von Maschinenbefehlen und Unterbrechungen.

Der Kernel des Überwachers kann nur über Unterbrechungen aufgerufen werden. Er läuft im Überwacherstatus

5 Unterbrechungsklassen:

- **Hardware - Fehler - Unterbrechungen**
- **E / A Unterbrechungen**
- **Externe Unterbrechungen**
- **Programm - Unterbrechungen**
- **System - Aufruf - Unterbrechungen (System Call, SVC)**

System - Aufrufe (System Calls) sind die einzige Möglichkeit für Benutzerprozesse, mit dem Überwacher zu kommunizieren.

Kernel
Supervisor

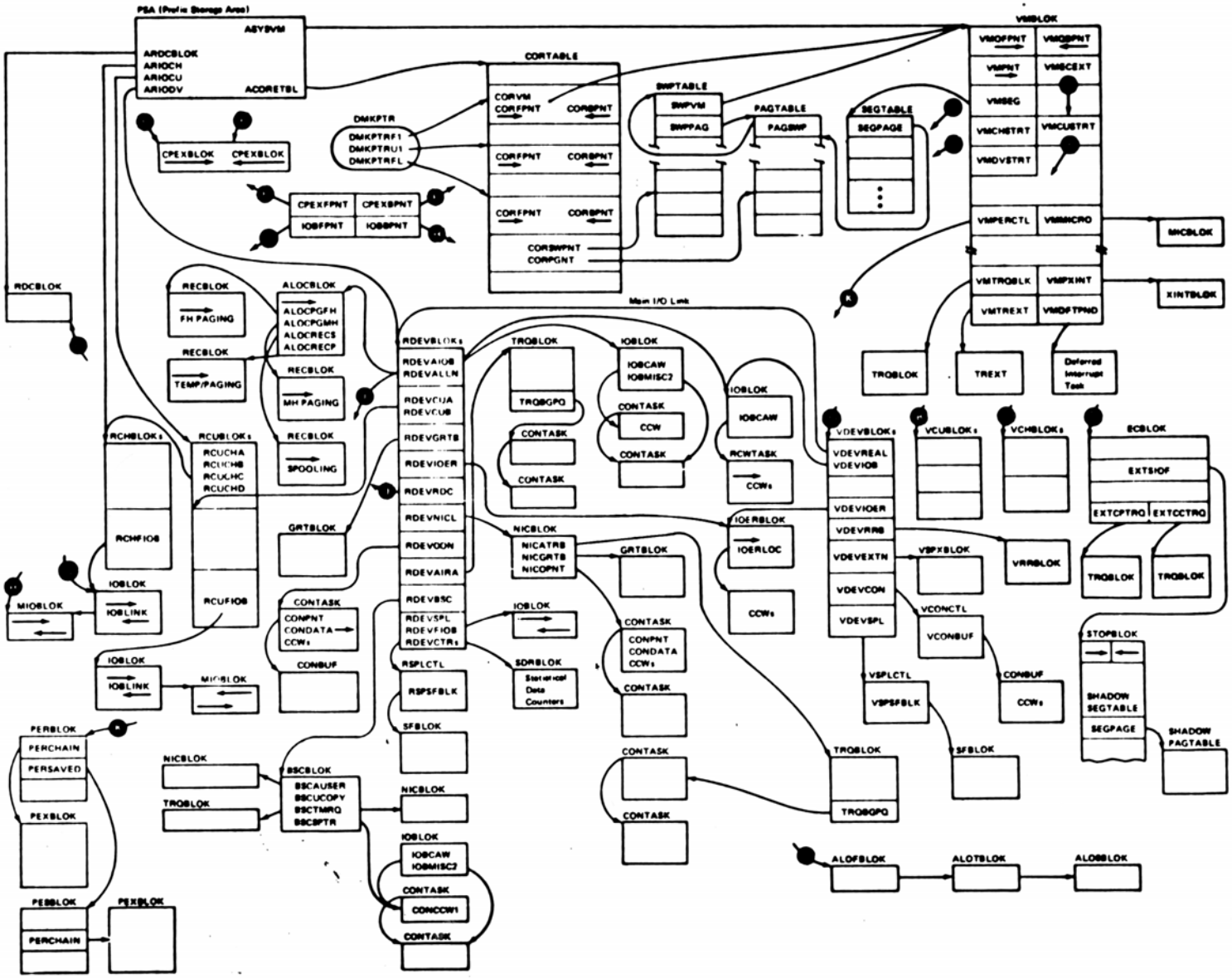
Der Kernel (Supervisor in z/OS Terminologie) besteht aus :

I. Datenbereichen (Control Blocks)

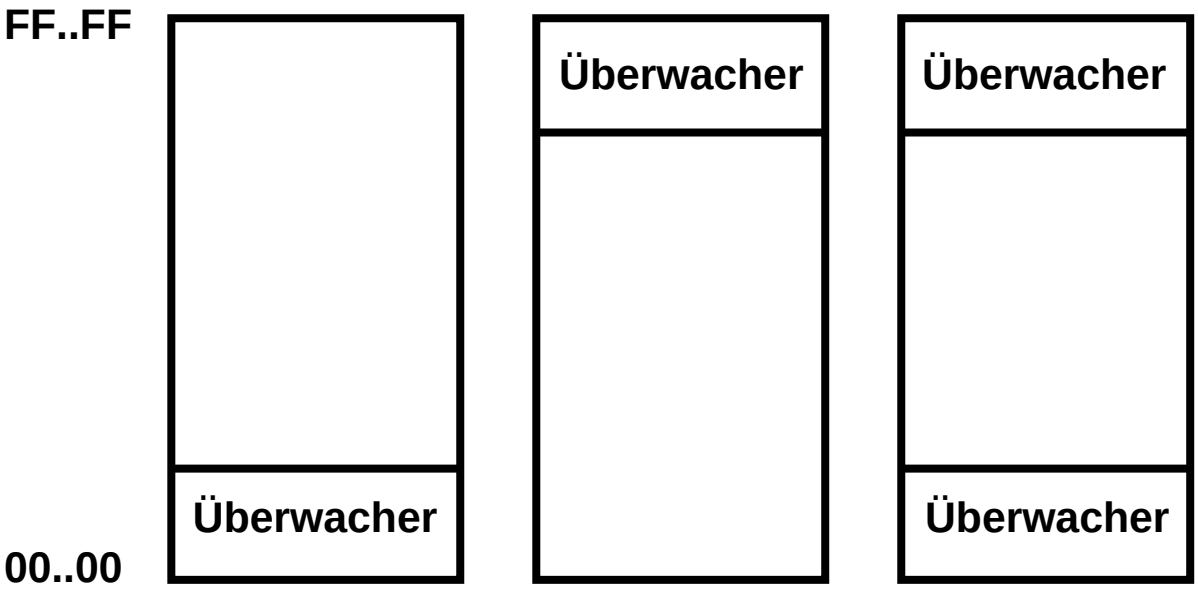
2. Programmteilen, welche Controlblock Daten manipulieren

In der Regel nicht strukturiert (Tanenbaum : The Big Mess)

Häufig sind 50 % der Ausführungszeit (Pfadlänge) eines Benutzer-Prozesses Überwacherfunktionen


PSA (Prefix Storage Area)
ASYSVM
ARDCBLOK
ARIOCH
ARIOCU
ARIODV
ACORETBL
CPEXBLOK
CPEXBLOK
DMKPTR
DMKPTRF1
DMKPTRU1
DMKPTRFL
CPEXFPNT
CPEXBPNT
IOBFPNT
IOBBPNT
CORTABLE
CORVM
CORFPNT
CORBPNT
CORFPNT
CORBPNT
CORFPNT
CORBPNT
CORSWPNT
CORPGNT
SWPTABLE
SWPVM
SWPPAG
PAGTABLE
PAGSWP
SEGTABLE
SEGPAGE
VMBLOK
VMIOFPNT
VMIOBPNT
VMPNT
VMBCEXT
VMSEG
VMCHSTRT
VMCUSTRT
VMDVSTRT
VMPERCTL
VMMICRO
VMTRQBLK
VMSPXINT
VMTREXT
VMDFTPND
MICBLOK
XINTBLOK
TRQBLOK
TREXT
Deferred Interrupt Task
RDCBLOK
RECBLOK
FH PAGING
RECBLOK
TEMP/PAGING
ALOCBLOK
ALOCPGFH
ALOCPGMH
ALOCRECS
ALOCRECP
RECBLOK
MH PAGING
RECBLOK
SPOOLING
Main I/O Limit
RDEVBLOKs
RDEVAIOB
RDEVALLN
RDEVCUA
RDEVCUB
RDEVGRTB
RDEVIOER
RDEVRDC
RDEVNICL
RDEVODN
RDEVAIRA
RDEVBSC
RDEVSPL
RDEVFIOB
RDEVCTRs
RSPLCTL
RSPSFBLK
SFBLOK
SDRBLOK
Statistical Data Counters
TRQBLOK
TRQBGPQ
CONTASK
CONTASK
CONTASK
IOBLOK
IOBCAW
IOBMISC2
CONTASK
CCW
CONTASK
IOBLOK
IOBCAW
RCWTASK
CCWs
IOERBLOK
IOERLOC
CCWs
NICBLOK
NICATRB
NICGRTB
NICOPNT
GRTBLOK
CONTASK
CONPNT
CONDATA
CCWs
CONTASK
CONTASK
CONTASK
TRQBLOK
TRQBGPQ
VDEVBLOKs
VDEVREAL
VDEVIOB
VDEVIOER
VDEVRRB
VDEVEXTN
VDEVCON
VDEVSPL
VCUBLOKs
VCHBLOKs
VRRBLOK
ECBLOK
EXTSHOF
EXTCPTRQ
EXTCCTRQ
TRQBLOK
TRQBLOK
VSPXBLOK
VCONCTL
VCONBUF
CCWs
CONBUF
VSPLCTL
VSPSFBLK
SFBLOK
STOPBLOK
SHADOW SEGTABLE
SEGPAGE
SHADOW PAGTABLE
RCHBLOKs
RCHFIOB
RCUBLOKs
RCUCHA
RCUCHB
RCUCHC
RCUCHD
RCUFIOB
GRTBLOK
MIOBLOK
IOBLOK
IOBLINK
IOBLOK
IOBLINK
MIOBLOK
CONTASK
CONPNT
CONDATA
CCWs
CONBUF
PERBLOK
PERCHAIN
PERSAVED
PEXBLOK
PERBLOK
PERCHAIN
PEXBLOK
NICBLOK
TRQBLOK
BSCBLOK
BSCAUSER
BSCUCOPY
BSCTMRQ
BSCPTR
NICBLOK
IOBLOK
IOBCAW
IOBMISC2
CONTASK
CONCCW1
CONTASK
ALOFBLOK
ALOTBLOK
ALOBBLOK

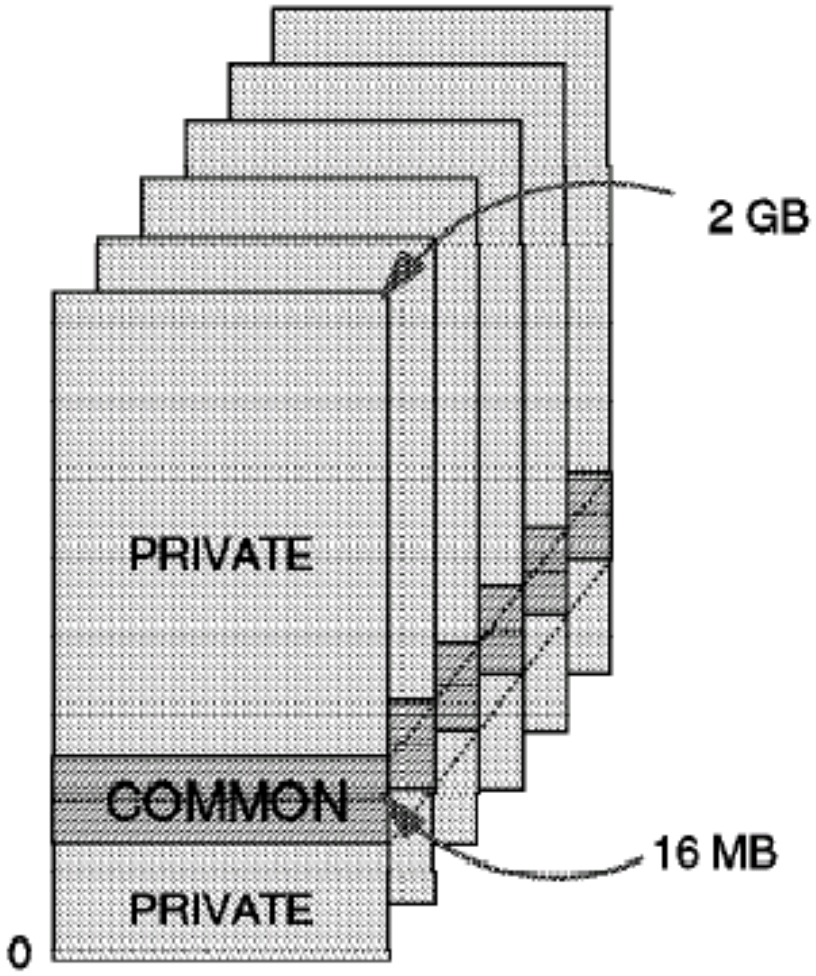
Kernel - Überwacher



Der Überwacher wird im untersten oder obersten Teil des Hauptspeicher - Adressenraums angeordnet (manchmal beides).

Dies kann der reale oder der virtuelle Adressenraum sein (manchmal beides).

Überwacherseiten sind gegen Zugriffe des Benutzerprogramms geschützt.



S/390 Address Spaces

Each address space represents a user in the system. An address space provides the capability for a program to address up to 2 GB of virtual storage minus the portion required for the z/OS or OS/390 system code and control blocks.

z/OS provides 64-bit virtual storage management support. The system area, called the common area, maps the executable z/OS or OS/390 code and the control blocks and work areas needed by all address spaces in the system. The common area is mapped around the 16 MB line.

S/390 Betriebssysteme

VSE	**IBM**	**mittelgroße Installationen**
VM	**IBM**	**Software Entwicklung, Unix Alternative**
z/OS, OS/390	**IBM**	**große Installationen**
TPF	**IBM**	**spezialisierte Transaktionsverarbeitung**
UTS 4	**Amdahl**	**based on System V, Release 4 (SVR4)**
OSF/1-M	**Hitachi**	**Open System Foundation Unix**
Linux	**Public Domain**	
BS2000	**Fujitsu/Siemens**	

Alle S/390 und zSeries Betriebssysteme (außer Linux) sind Server Betriebssysteme, optimiert für den Multi-User Betrieb

I5/OS (früher OS/400) ist ein IBM Betriebssystem, welches suf der PowerPC Plattform läuft. Es adressiert ähnliche Anforderungen wie die S/390 Betriebssysteme. Wegen fehlender Skalierungseigenschaften vor allem in mittelständischen Betrieben eingesetzt,

S/390 Betriebssysteme
Lizenzen weltweit
1999

z/OS, OS/390, MVS	**13 500**
VSE	**12 000**
TPF	**300**
UTS 4 Amdahl Unix SVR4	**300**
zLinux	

Gastbetriebssysteme

VM	**10 000**

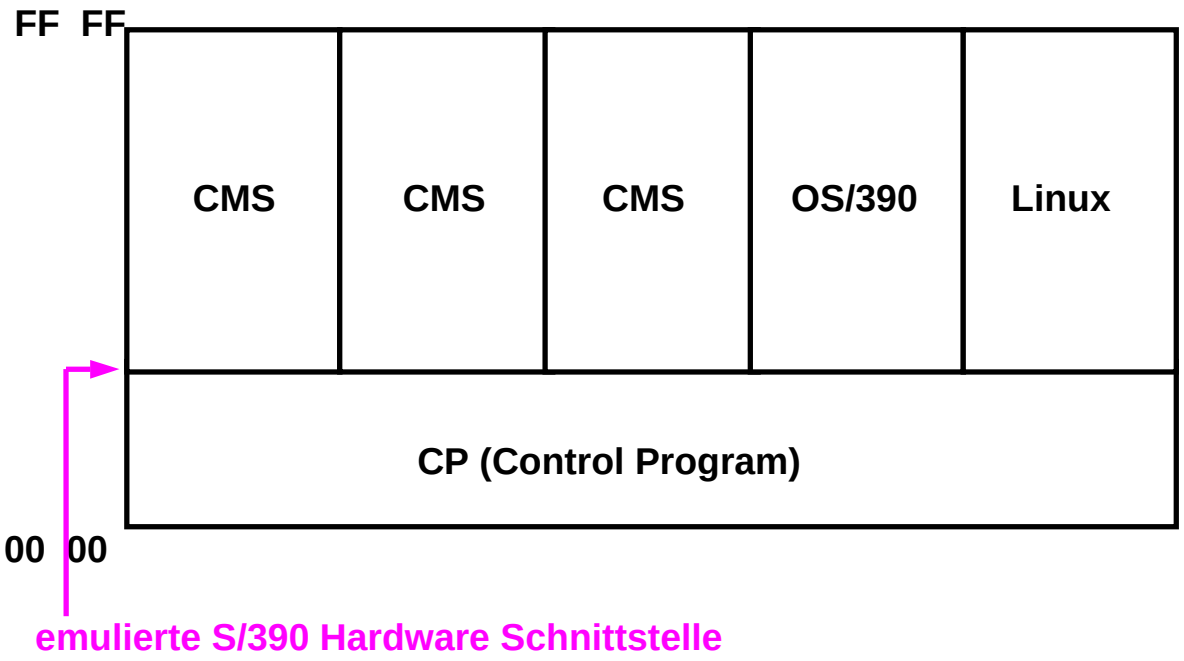
Die meisten der VM Installationen laufen auf Rechnern, auf denen VSE oder OS/390 als Haupt-Betriebssystem installiert ist.

TPF
Transaction Processing Facility

American Airlines Sabre Group

30 000 Reisebüros
3 Mill. registrierte Einzelkunden
400 Fluggesellschaften
50 Mietwagenfirmen
35 000 Hotels
Eisenbahngesellschaften
Fähren
Kreuzfahrten

bis zu 15000 Transaktionen / s



zVM Betriebssystem

CP läuft im Überwacherstatus.

CMS und alle anderen Gast Betriebssysteme (einschließlich ihrer Kernel Funktionen) laufen im Problemstatus. Privilegierte Maschinenbefehle (z.B. E/A) werden von CP abgefangen und interpretativ abgearbeitet.

Volle S/390 Kompatibilität für alle Gast Betriebssysteme. Geringer Performance Verlust (< 5 %).

CMS (Conversational Monitor Program) ist ein besonders für die Software Entwicklung ausgelegtes Einzelplatz Betriebssystem. Für 1000 gleichzeitige CMS Benutzer werden 1000 CMS Instanzen angelegt. Ähnliches ist mit Linux/390 möglich.

Plattenspeicherplatz wird allen Gastbetriebssystemen in der Form virtueller “Minidisks” statisch zugeordnet. Hauptspeicherplatz wird dynamisch verwaltet.

z/OS und OS/390

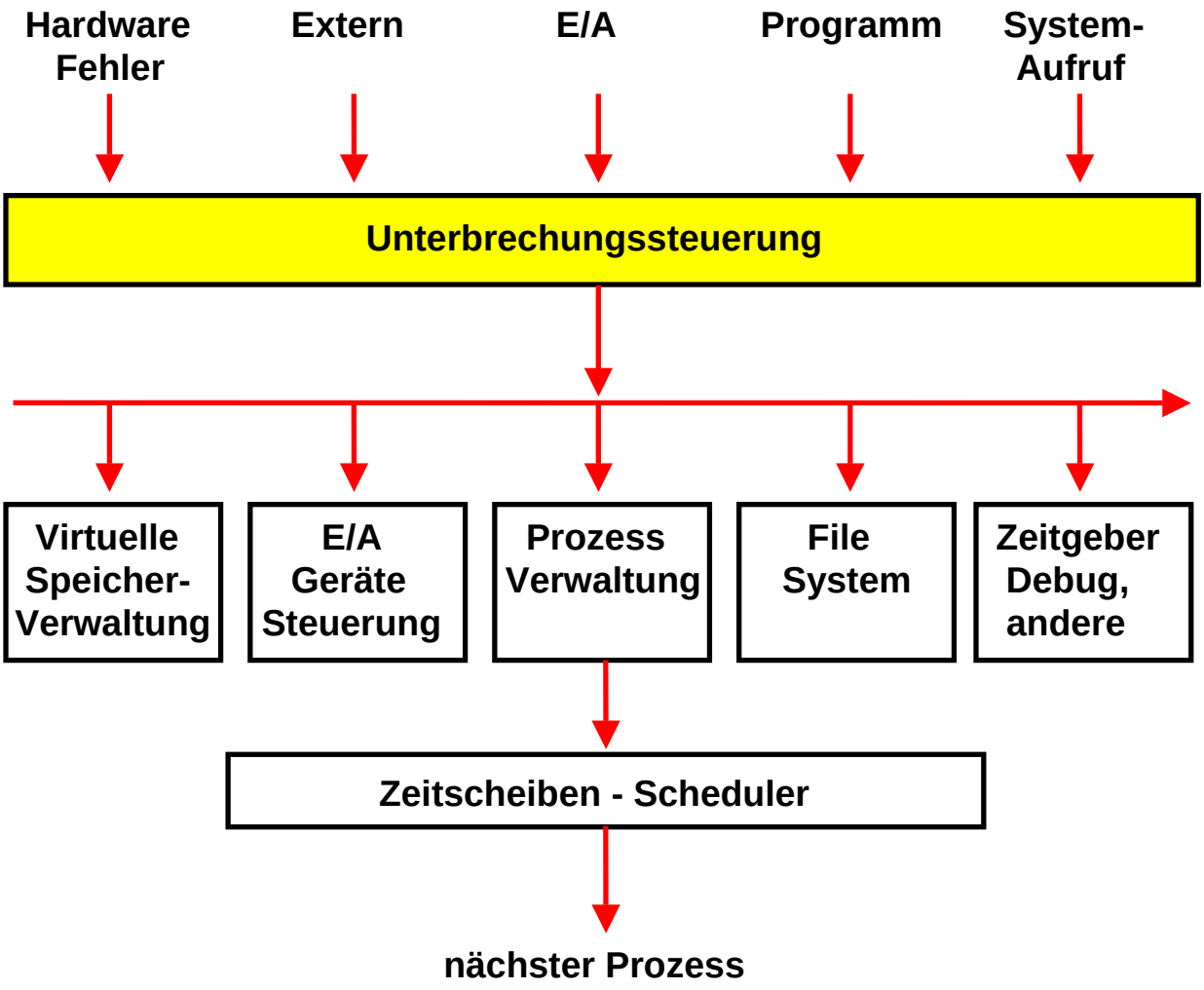
Anfang 1966 als OS/360
(reines Stapelverarbeitungssystem) eingeführt
- Fred Brooks

Spätere Namenswechsel
OS/360 → MFT → MVT → MVS

1996 Namensänderung MVS nach OS/390 –

- **Bündeln von mehr als 70 Komponenten**
- **Vereinfachung der Installation und der Wartung**

2000 Namensänderung OS/390 nach z/OS
64 Bit Adressierung



Struktur des Überwachers

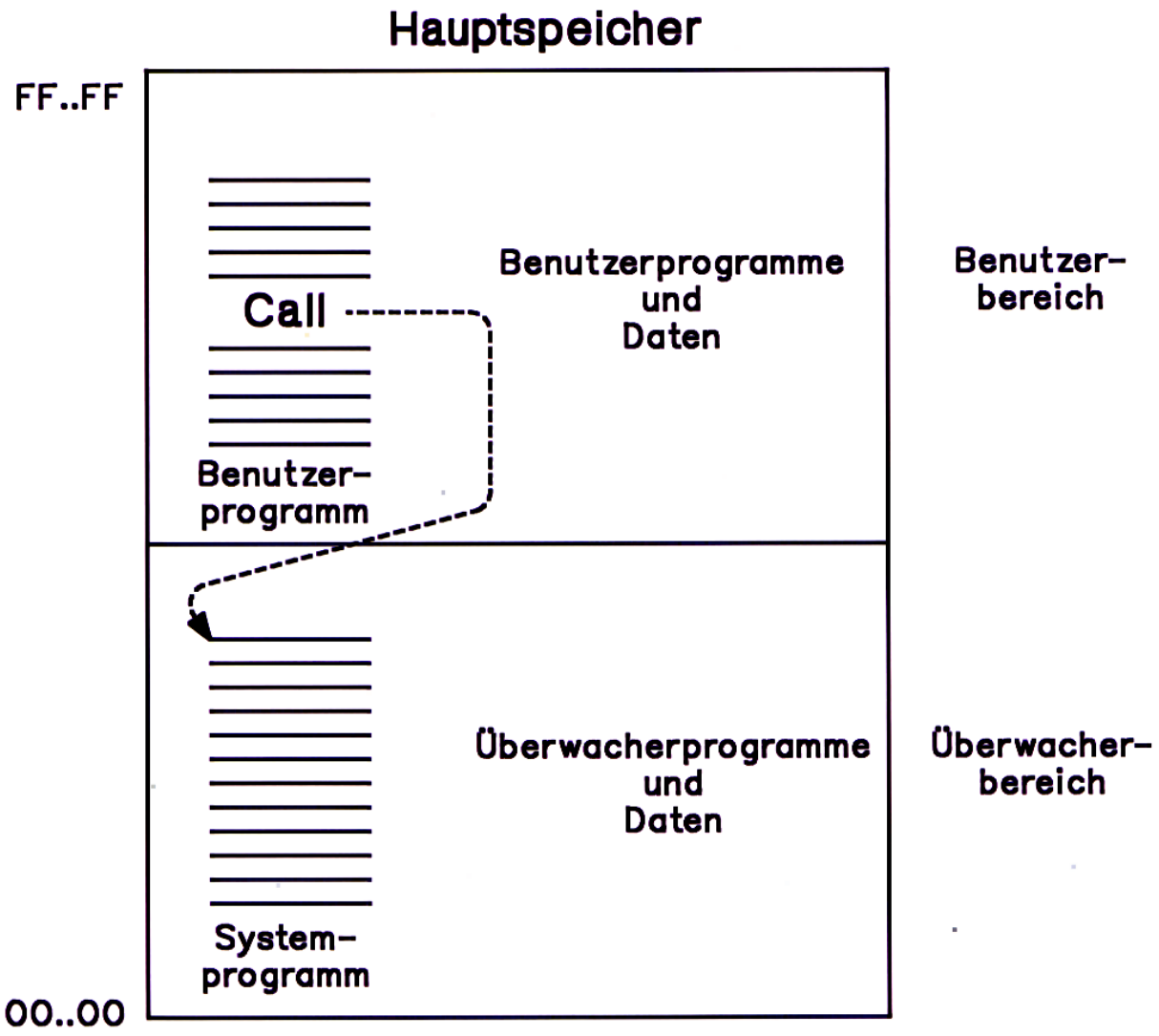
Der Ufruf des Überwachers (Supervisor, Kernel) erfplgt grundsätzlich über Unterbrechungen.

Je nach Art der Unterbrechung werden unterschiedliche Komponenten des Überwachers aufgerufen.

Benutzerprozesse nehmen Dienste des Überwachers über eine architekturierte Schnittstelle, den Systemaufruf /System Call, Supervisor Call, SVC –) in Anspruch. Diese Begriffe haben alle die gleiche Bedeutung.

Der Scheduler (Zeitscheibensteuerung) sucht den nächsten auszuführenden Prozess aus.

Unterbrechungssteuerung



Systemaufruf
(System Call)

Auswahl von System Calls (Beispiel UNIX)

Prozess-Kontrolle

- **Beenden (exit)**
- **Erzeugen (fork), Beenden (kill)**
- **Warten auf Signal (pause)**
- **Warten auf Prozeßende (wait)**
- **Ausführen (exec)**
- **Kommunikation etablieren (pipe)**
- **Priorität ändern (nice)**

Datei-Behandlung

- **Erzeugen (creat)**
- **Öffnen (open), Schließen (close)**
- **Lesen (read), Schreiben (write)**
- **Repositionieren des Zeigers (lseek)**
- **Verbinden mit Katalog (link)**
- **Lösen von Katalog (unlink)**
- **Dateiattribute ermitteln, ändern (fstat, chmod, chown, ...)**

Geräte-Behandlung (analog Datei, z.B.)

- **Steuerfunktion ausführen (ioctl)**

Informationsverwaltung

- **Zeit ermitteln, setzen (time, stime)**
- **Benutzer-, Prozeßverwaltung (getuid, getpid)**

/usr/lib/libc.a:

abs.o:
errno.o:
ctype.o:
string.o:
abort.o:
bsearch.o:
strtol.o:
atoi.o:
rand.o:
div.o:
isatty.o:
strerror.o:
getenv.o:
atof.o:
ctime.o:
qsort.o:
system.o:
strftime.o:
tzset.o:
ftime.o:
closedir.o:
opendir.o:
readdir.o:
rewinddir.o:
fgetgrent.o:
getgrent.o:
getgrgid.o:
getgrnam.o:
grpopen.o:
grpread.o:
initgroups.o:
atexit.o:
bcopy.o:
exit.o:
perror.o:
tmpnam.o:
writev.o:
sig_restore.o:
setjmp.o:
getcwd.o:
mktemp.o:
popen.o:
getopt.o:
sigmask.o:
tcattr.o:
sysconf.o:
sleep.o:
tcsetpgrp.o:
rename.o:
ttyname.o:
cfsetget.o:
tcflow.o:
fgetpwent.o:
getpw.o:
getpwent.o:
getpwnam.o:
getpwuid.o:
putpwent.o:
pwdopen.o:
pwdread.o:
clrerr.o:
doprnt.o:
doscan.o:
fdopen.o:
fgetc.o:
fgets.o:
filbuf.o:
findiop.o:
flsbuf.o:
fopen.o:
fprintf.o:
fputc.o:
fputs.o:
fread.o:
freopen.o:
fseek.o:
ftell.o:
fwrite.o:
getchar.o:
gets.o:
getw.o:
printf.o:
putchar.o:
puts.o:
putw.o:
rew.o:
scanf.o:
setbuf.o:
setbuffe.o:
sprintf.o:
ungetc.o:
vfprintf.o:
vprintf.o:
vsprintf.o:
setvbuf.o:
_exit.o:
access.o:
acct.o:
alarm.o:
brk.o:
chdir.o:
chmod.o:
chown.o:
chroot.o:
close.o:
creat.o:
dup.o:
dup2.o:
exec.o:
execp.o:
execve.o:
fcntl.o:
fork.o:
fstat.o:
getegid.o:
getpgrp.o:
ioctl.o:
kill.o:
link.o:
lseek.o:
mkdir.o:
mknod.o:
mount.o:
nice.o:
open.o:
pause.o:
pipe.o:
read.o:
rmdir.o:
setgid.o:
setpgrp.o:
setsid.o:
setuid.o:
signal.o:
stat.o:
stime.o:
sync.o:
time.o:
times.o:
ulimit.o:
umask.o:
umount.o:
uname.o:
unlink.o:
ustat.o:
utime.o:
wait.o:
write.o:
setreuid.o:
setregid.o:
sigsusp.o:
sigpend.o:
sethostnam.o:
gethostnam.o:
setrlimit.o:
getrlimit.o:
getrusage.o:
gettime.o:
settime.o:
getgroups.o:
geteuid.o:
getgid.o:
getpid.o:
getppid.o:
getuid.o:
setgroups.o:
select.o:
symlink.o:
lstat.o:
readlink.o:
uselib.o:
gmalloc.o:
ldexp.o:
modf.o:
readv.o:

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

System call	Meaning
creat(name, mode)	Create a file; *mode* specifies the protection mode
unlink(name)	Delete a file (assuming that there is only 1 link to it)
open(name, mode)	Open or create a file and return a file descriptor
close(fd)	Close a file
read(fd, buffer, count)	Read *count* bytes into *buffer*
write(fd, buffer, count)	Write *count* bytes from *buffer*
lseek(fd, offset, w)	Move the file pointer as required by *offset* and *w*
stat(name, buffer)	Return information about a file
chmod(name, mode)	Change the protection mode of a file
fcntl(fd, cmd, ...)	Do various control operations such as locking (part of) a file

Die wichtigsten Unix System Calls für die Verwaltung von Dateien

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

File management

Call	Description
fd = open(file, how, ...)	Open a file for reading, writing or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

Übersicht über die wichtigsten Unix Kommandos

Directory and file system management

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

Miscellaneous

Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

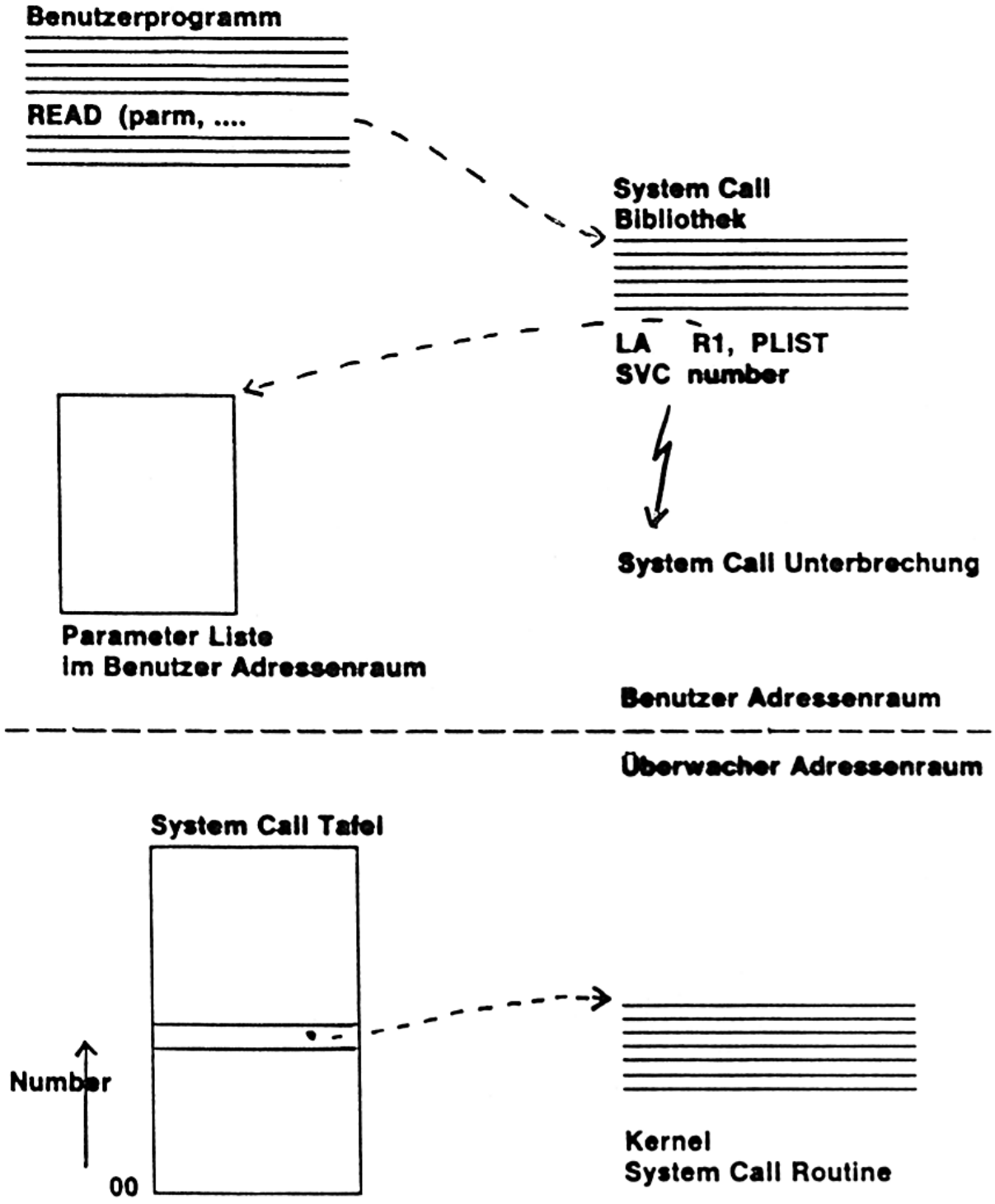
cat f	List contents of file
cat f1 f2 >f3	Concatenates f1(file 1) & f2(file 2) into f3(file 3)
cd	returns you to your home or main directory
cd /	takes you to the root, as far up (to the left) as far as possible
cd	to move down (right in the pathname) a directory
cd ..	moves you up (left in pathname) a directory; likewise,
cd ../../..	moves you up (left in the pathname) 3 directory levels
chmod ###	changes your protections. The order is: you|group|universe (rwxrwxrwx). There will be either a d or - before it. If there's a d, then it's a directory. If there's not, then it's a file. You set the protections in the order rwx (read=1, write=2, execute=4). So, to set the protections for the directory directoryname: you rwx, group r-x, universe r--, you would enter: chmod 751 .
clear	to clear screen
compress	compresses the file filename and puts a .Z extension on it. To uncompress it, type uncompress
cp f1 f2	Copy file f1 into f2
cp -r	copies the directory directoryname and renames it newdirectoryname
^-c (ctrl-c)	to kill a running process
^-d (ctrl-d)	to close an open window
df	gives disk usage
diff f1 f2	Lists file differences
dig host	domain name, IP address, and alias information for the given host.
dosdir	to do a "dir" (~ls in UNIX) on a DOS floppy in the disk drive
dosread	to read a file from a DOS floppy to your computer account
doswrite	to write a file from your computer account to a DOS floppy
du	lists all subdirectories and their sizes (in blocks?) and total directory size (in blocks?) (takes a long time)
du -a	lists all files and their sizes (in blocks?) in present directory and total directory size (in blocks?) (takes a long time)
du -s	lists overall directory size (in blocks?) (long but clean)
env	shows current environment set-up
find	Searches the named directory and it"s sub-directories for files. Most frequently called like this: `find ./ -name "t*" -print` Which searches the current directory (and all of its sub-directories) for any files that begin with the letter "t" and

	then prints them out. If you are looking for a specific filename, then replace "t*" with "filename", and "find" will print out all incidences of this file.
finger @.	(e.g., finger johndoe@ksu.edu fingers Johndoe at Kent State University)
ftp	establishes an ftp link with machinename
gzip	produces files with a .gz extension.
gunzip	decompress files created by gzip, compress or pack.
ispell f	Interactively checks the spelling of the file f, giving logical alternatives to the misspelled words. Type "?" to get help. "ispell" can be accessed from the command line, and also through emacs with M-x ispell-buffer.
kill -9 -1	(from a remotely logged-in site) kills all running processes (essentially forces a logout) *not to be used unless nothing else works* kill -9 process-id# - kills a running process
lpq	shows UNIX print queue
lpr	to print the file
lpqrm job#	removes job from printer queue
ls	shows listing of files in present directory
ls -a	shows listing of all files in present directory
ls -l	shows long listing of files in present directory
ls -la | more	shows long listing of all files in present directory
man command	shows help on a specific command.
mkdir	creates a new directory called directoryname
more	to view the contents of a file without making changes to it one screen at a time. Hit q to quit more.
mv f1 f2	Rename file f1 as f2
mv	moves the file called filename to the directory directoryname
nslookup host	domain name, IP address, and alias information for the given host. e.g., nslookup www.kent.edu gives related data for www.kent.edu
passwd	to change your password (takes an hour or so to take effect on all machines)
ping host	to test if the host is up and running.
pwd	present working directory
ps	Shows processes running
ps -flu	Shows detailed description of processes running

pquota	Shows printer quota
quota -v	Shows current disk usage and limits.
rlogin	allows you to remotely log in to another machine on which you have access privileges
rm f	Delete(removes) the file f.
rm - r	removes the directory named directoryname
s f	Alphabetically sort f.
talk	establishes an e-talk session with user@machinename
tar	combines multiple files into one or vice-versa
telnet	allows you to remotely log in to another machine on which you have access privileges
uncompress	uncompresses filename.Z
users	shows who's logged in on the machine
vi	to open the file called filename in the vi text editor
who	Shows who is currently logged on the system.
whoami	shows username of person logged in that window
whois domain_name	lists the domain registration record, e.g., whois kent.edu will produce the domain record for kent.edu
*	wild card character representing any # or characters
date	shows the time and date
date -u	shows grewich mean time
.	a short cut that stands for the location you are at in a pathway. ex. cp (file (though a pathway) (. (the location you are at)
..	move to parent directory from any comand ex. mv (file name) .. or cd .. etc.
pwd	shows where you are in the pathway
?	wild card character representing one character, can be used in succesion
~	abbreviation for the home file ex. ls ~ lists files in home dir w/o moving there
zip	best compression for IBM files.

BEISPIELE für z/OS und S/390
SYSTEM AUFRUF MAKROS

ATTACH
RESET
EXIT
ABEND
OPEN
CLOSE
TGET
TPUT
ALLOCATE
DYNALLOC ()
DEALLOCATE ()
GETMAIN 2-92
FREEMAIN
VTAM
WTO (write to operator)
WTOR (write to operator, reply)



Ablauf des Systemaufrufes

In der Parameter Liste wird der System Call Routine eine Reihe von Parametern zur Verfügung gestellt, z.B. die Datei Definition beim Open System Call.

Systemaufruf - Betriebssystem-Funktionen

In der Umgangssprache bezeichnet man als Systemaufrufe häufig die Zugriffe zu Funktionen in einer System Call Makro - Bibliothek

> **z.B. C-Lib für den C-Compiler, oder eine System Call - Bibliothek, die von allen Sprachübersetzern gemeinsam benutzt werden kann**

Sie bestehen aus einem Funktionsnamen und einer Parameter-Liste

Beispiel (UNIX, C) :

```
count = read (file, buffer, nbytes)
```

Die Bibliotheks-Routine läuft im Benutzer-Status.
Sie verwendet einen Systemaufruf-Maschinenbefehl, der die Umschaltung in den Überwacherstatus vornimmt (Systemaufruf-Unterbrechung, Beispiel SVC unter z/OS).

Als Teil der Hardware-Architektur sind Systemaufruf-Maschinenbefehle präzise definiert. Sie verursachen Systemaufruf-Unterbrechungen.

0A	I

SVC Maschinenbefehl
S/390 und zSeries

SVC hat die Wirkung einer Unterbrechung.

Der Inhalt von „I“ wird im Hauptspeicher auf der Adresse 0139D abgespeichert.

Der SVC First Level Interrupt Handler benutzt den Inhalt der realen adressen (01360139) um in die gewünschte Systemaufruf Routine des Überwachers zu verzweigen

(zArchitecture Principles of Operations, p. 6-47)

Intel IA32 Architektur
“INT” Maschinenbefehl

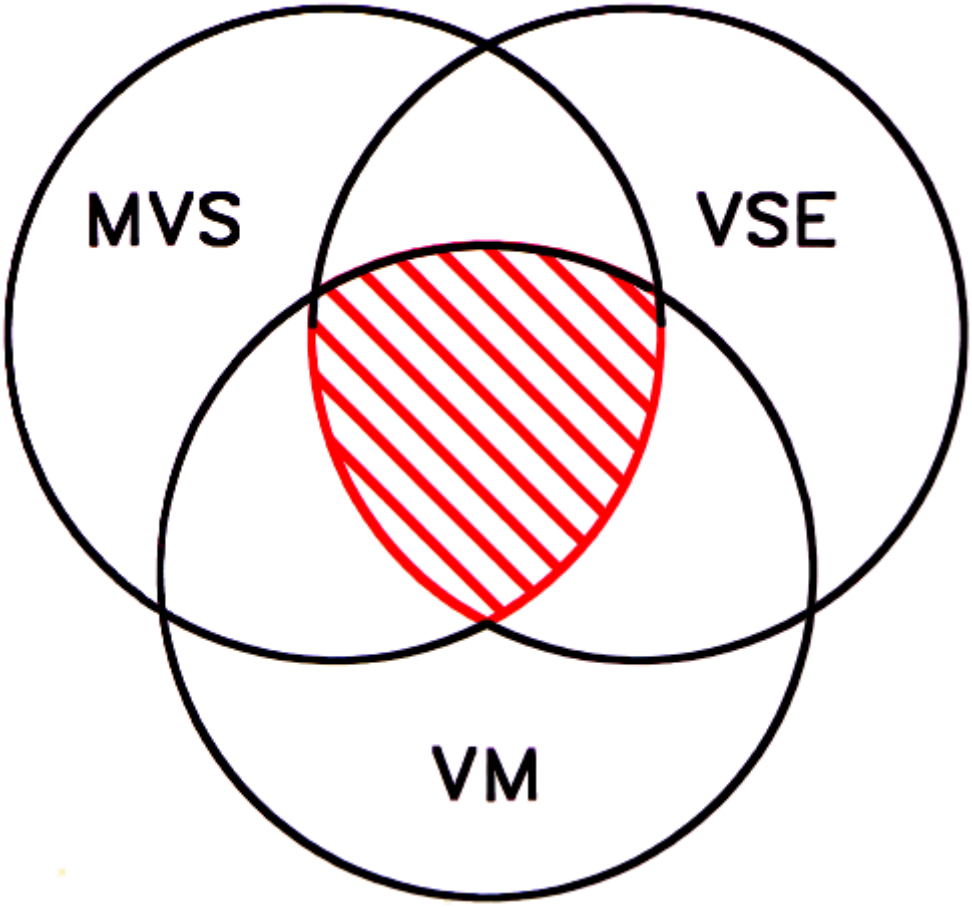
INT verhält sich ähnlich wie der OS/390’s SVC Maschinenbefehl.

Status is saved in a predefined place, control passes in kernel mode to a handler defined by the operating system and the handler gives control to a routine designated by the operand of the instruction.

Unterschied zwischen

Systemaufruf Maschinenbefehle
Beispiel SVC

Systemaufruf Betriebssystemfunktionen
Beispiele OPEN, WRITE



Gemeinsame Untermenge von Systemaufrufen für die z/OS, z/VM und VSE Betriebssysteme

Kommando Interpreter
(Shell)

ist dasjenige Anwendungsprogramm, welches es dem Benutzer ermöglicht, den Rechner zu steuern.

Beispiele Für Kommandos:

Display Files (z. B. FILEL, DIR, ls -als,)

Copy

Compare

Move

Edit (ISPF editor, vi editor)

usw.

Standard z/OS Shessls:

- **TSO (mit ISPF)**
- **Unix System Services (USS) Shells**

11.04.2008 10:06 heise online
<< Vorige | Nächste >>

IT-Analysten: Windows kollabiert Meldung vorlesen

Windows ist in Schwierigkeiten – und damit auch Microsoft. Das meinen zumindest Analysten des IT-Marktforschungsinstituts Gartner, die auf der Konferenz Gartner Emerging Trends mit Microsoft und seinen Bemühungen um Windows hart ins Gericht gingen. Die Entwicklungszyklen seien viel zu lang und brächten nur begrenzte Innovationen in neuen Windows-Versionen, lautet einer der Kritikpunkte laut einem Bericht der PC World. Und laut Computer World beschrieben die Gartner-Analysten die Situation als unhaltbar: Windows kollabiere, Microsoft müsse grundlegende Änderungen an seinem Betriebssystem bewerkstelligen, anderenfalls sei der Konzern weg vom Fenster: Windows, so wie man das heute kenne, müsse ersetzt werden.

Die harsche Kritik trifft alle Bereiche von Microsoft, Entwickler, Verkäufer, Marketiers und natürlich das Management: Microsoft habe nicht auf den Markt reagiert und sei überlastet mit Code-Altlasten aus zwei Jahrzehnten; dies sei einer der Gründe, warum Vista so wenig wirkliche Neuerungen liefere und bei den Kunden nicht besonders gut ankomme. Außerdem gebe es mittlerweile ernsthafte Konkurrenz auf einer ganzen Reihe von Märkten – zuletzt Anfang Februar hatte Gartner etwa der Open Source eine rosige Zukunft attestiert. Dies mache Windows irrelevant, falls Microsoft nicht gegensteuere; auch Webanwendungen und kleinen, auf bestimmte Funktionen ausgerichtete Geräte grüben Microsoft das Wasser ab. Microsoft habe zwar für alle Plattformen System-Versionen, diese seien aber nicht wirklich zueinander kompatibel; sie böten Anwendern, die spezialisierte Geräte mittlerweile völlig unabhängig von der Systemplattform einsetzten, wenig Vorteile.

Als Abhilfe sehen die Gartner-Analysten eine weit stärker modularisierte Version von Windows als Vista an, bei dem Microsoft ja bereits seine Modularität betont. Auch könne in Virtualisierung von Systemen und Systembestandteilen, etwa für Abwärtskompatibilität mit älteren Anwendungen, eine Lösung liegen. Schnelle Änderungen seien bei Microsoft angesichts des Milliarden Dollar schweren Windows-Geschäfts nicht zu erwarten. (jk/c't)

PC World

Gartner Explains Why Windows Is Broken

Esther Schindler, CIO.com
Thursday, April 10, 2008 4:00 PM PDT
Recommend this story?

In a session at the Gartner Emerging Trends conference this week, analysts Neil MacDonald and Michael Silver identified many reasons that Windows (and thus Microsoft) are in trouble.

Microsoft's operating system (OS) development times are too long and they deliver limited innovation; their OSs provide an inconsistent experience between platforms, with significant compatibility issues; and other vendors are out-innovating Microsoft. That gives enterprises unpredictable releases with limited value, management costs that are too high, and new releases that break too many apps and take too long to test and adopt. With end users bringing their own software solutions into the office... well, it's just a heck of a sad story for Microsoft.

Those arguments probably don't surprise you. (See Should Microsoft Throw Away Vista? and Vista Never Had Its Moment in 2007.) But the Gartner analysts offered several more points to show how Windows is in a whole new world of hurt. High on the list is Windows' complexity, its lack of modularity, its hardware footprint (particularly on low-end PCs), and the increasing movement to Web-based and other OS-agnostic applications.

A few of their arguments:

Mature markets have limited growth in terms of PC hardware. The computer hardware business is expected to grow only 2 percent to 8 percent between 2005 and 2011. The opportunities for PCs are higher in emerging markets, where the growth rate is 16-24 percent for PC hardware-but they're more price-sensitive so vendors and enterprises have to keep the price down. That means less memory and storage, for example-and Vista is not appropriate for that sort of memory model. Linux is the preferred OS on low end PCs including "one laptop per child" and certainly Microsoft doesn't want to see that happen. "All these things are in opposition to what we've seen with people expanding PC use year after year," MacDonald said.

Version compatibility is relevant in more than software development terms. For example, they said, iPhone's version of OS X is closer to the desktop version of the Mac OS than Windows Mobile is to Vista.

Servers are evolving in multiple and sometimes conflicting directions. Some industry trends imply that we need to scale up computing, such as single instance data stores and partitioning. Others are driving it down, such as grid and cloud computing, server farms and cluster computing. The result, they believe, is that enterprises will want we want to customize the OS based on the need.

Microsoft has taken some first steps in this regard, they pointed out; for example, Windows Server 2008 can be preconfigured based on roles. "That's a step in the right direction, but it's still fairly superficial," said Silver. What's needed is a radical change in architecture that goes beyond packaging DLLs, he added.

The move to server-agnostic applications is still in its infancy but will soon have a major effect on enterprise computing. The legacy applications won't go away, even if the exciting stuff is being done on Internet-based apps, they said. But it won't stay that way. Today, 70 to 80 percent of corporate applications require Windows to run, but the Gartner analysts expect a tipping point in 2011, when the majority of these applications will be OS-agnostic, such as Web applications. "Sometime in the middle of the next decade, Windows will be playing a much less important role on the desktop," MacDonald said.

Virtualization changes our view of what operating systems are. Virtualization starts offering levels of abstraction between the OS and the hardware, pointed out the analysts. The hypervisor is taking on some of the role of what the OS did. "Is this the time to redraw some of these lines?" asked Silver. "For us in IT, the interjection of these new layers helps introduce fluidity, and lets us better manage IT."