Einführung in z/OS und OS/390

Dr. rer. nat. Paul Herrmannn
Prof. Dr.-Ing. Wilhelm G. Spruth

WS 2006/2007

Teil 3

z/OS Betriebssystem

S/390 Betriebssysteme

VSE	**IBM**	**mittelgroße Installationen**
VM	**IBM**	**Software Entwicklung, Unix Alternative**
z/OS	**IBM**	**große Installationen (OS/390, MVS)**
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**	

Alle S/390 Betriebssysteme sind Server Betriebssysteme, optimiert für den Multi-User Betrieb

S/390 Betriebssysteme
Lizenzen weltweit

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

Gastbetriebssysteme

VM	**10 000**
Linux	

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 10 000 Transaktionen / s

Amadeus System der Deutschen Lufthansa

Transaction Processing Facility TPF

13. Oktober 2006. Die Firma Worldspan, ein weltweiter Anbieter von Reise-Reservierungssystemen, hat sich für den Einsatz von sechs IBM System z9 Enterprise Class (EC) Mainframe-Servern entschieden. Worldspan will damit sein Angebot an elektronischen Datendiensten erweitern, um circa 700 Anbietern von Reiseangeboten und Millionen von Reisenden weltweit eine gemeinsame Plattform anbieten zu können.

Worldspan setzt die neuen IBM System z9 EC Server ein, um sowohl Reisebüros als auch Anbietern von Online-basierten Reisediensten die Möglichkeit zur Nutzung des weltweiten Global Distribution System (GDS) zu geben, über das zum Beispiel die Bestellung und Buchung von Reiseprodukten von Flugzeugtickets, Hotels, Mietwagen und andere Reisedienstleistungen durchgeführt wird.

Durch die Nutzung der Software „IBM Transaction Processing Facility“ (TPF) ist Worldspan in der Lage, 17.000 Kundenanfragen pro Sekunde auf den Mainframes zu beantworten.

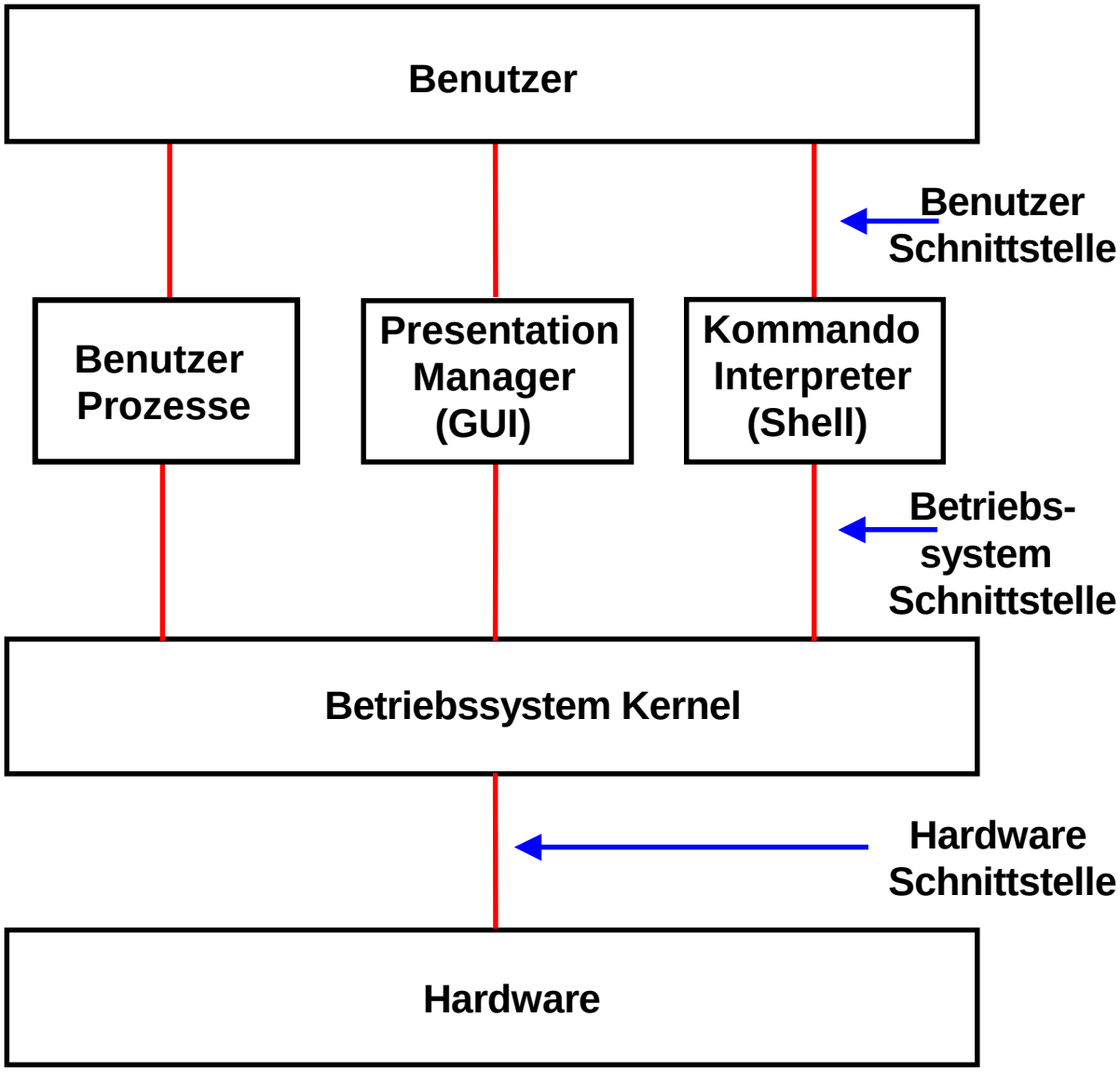
z/OS und OS/390

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

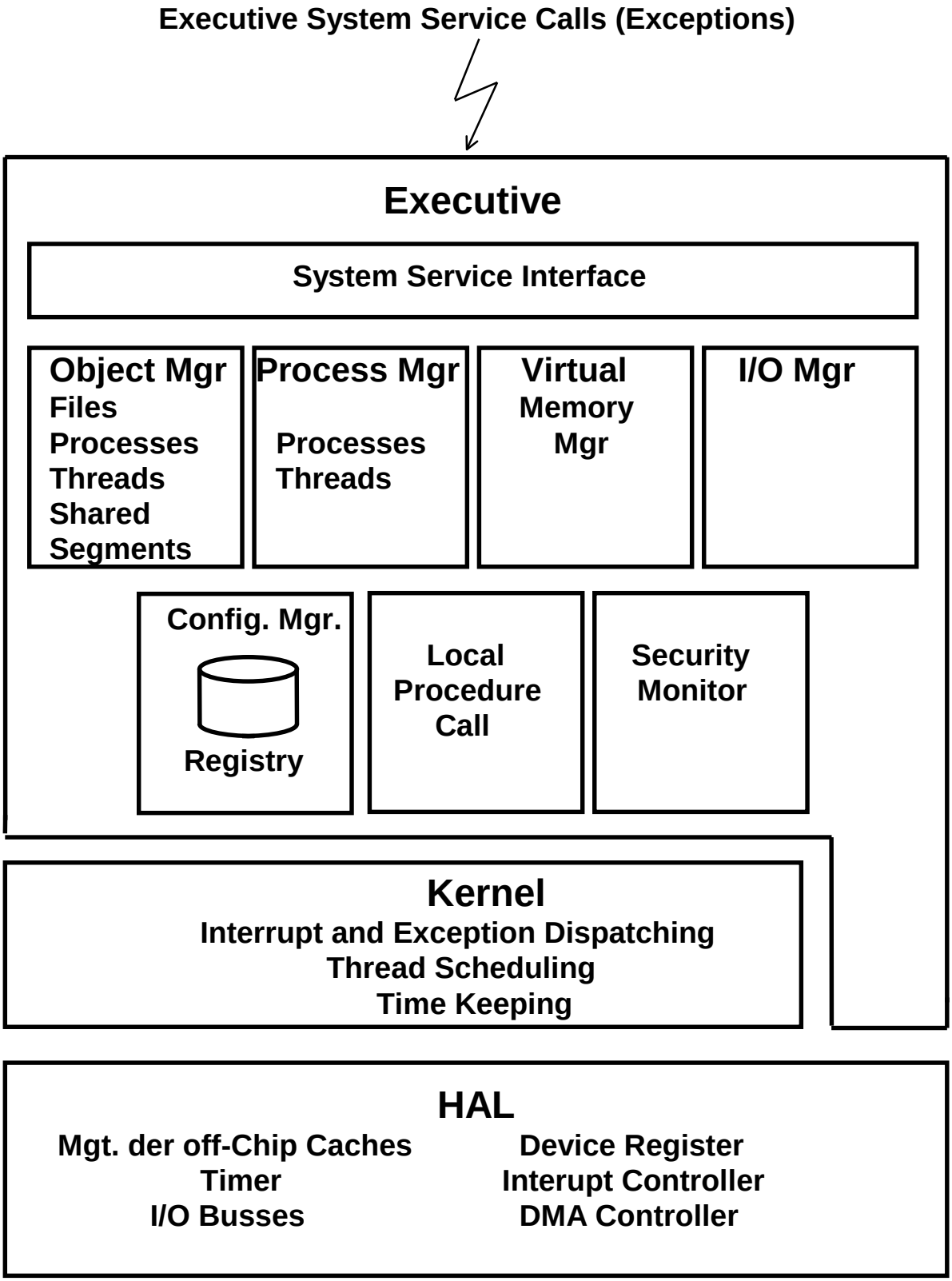
Spätere Namenswechsel ~~OS~~/360 ~~MFT~~ → MVT → MVS

1996 Namensänderung MVS nach OS/390 - Bündeln von mehr als 70 Komponenten: Vereinfachung der Installation und der Wartung

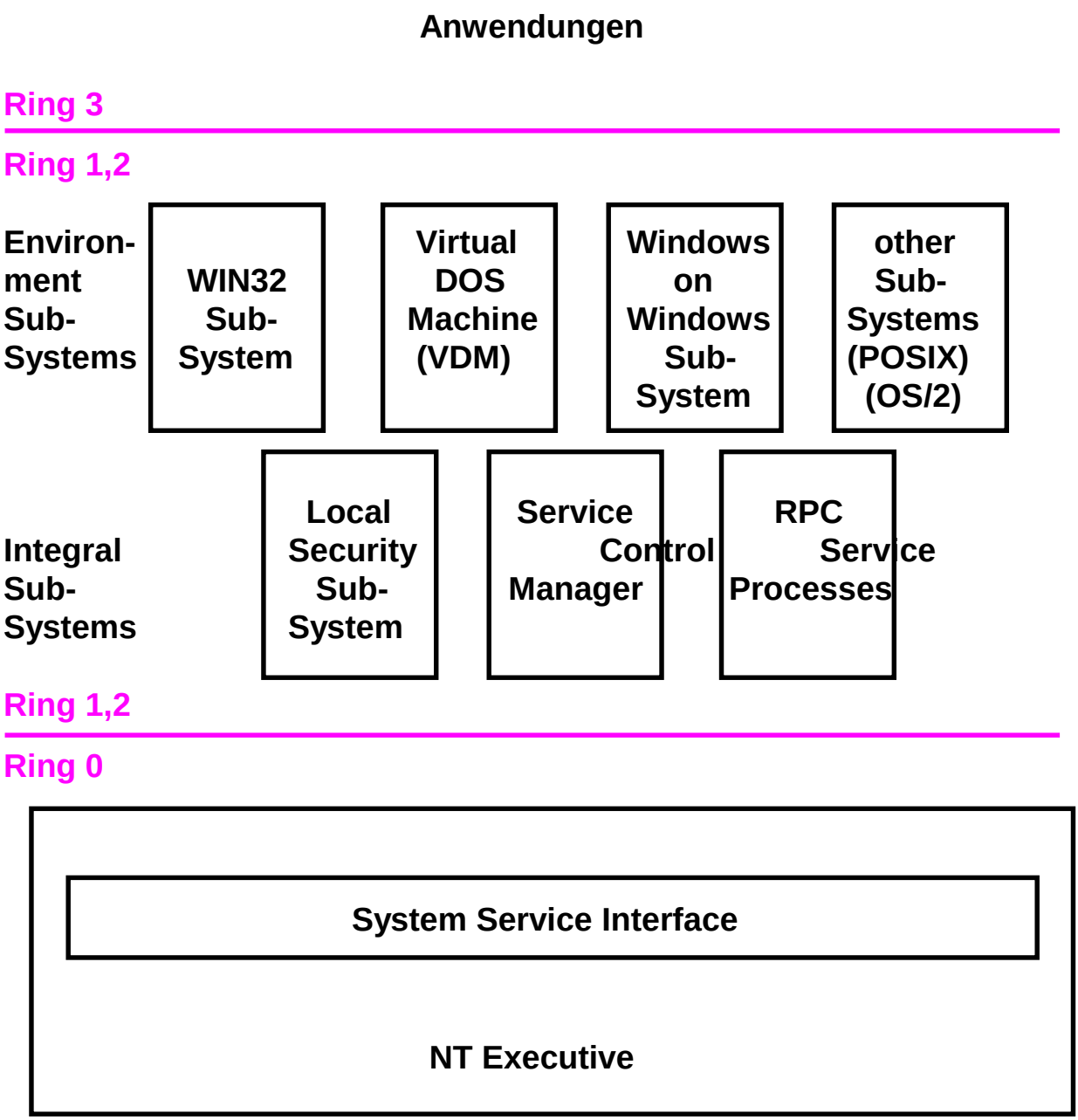
2000 Namensänderung OS/390nach z/OS - 64 Bit Adressierung



Schichtenmodell der Rechnerarchitektur
Windows, Linux, Unix, z/OS

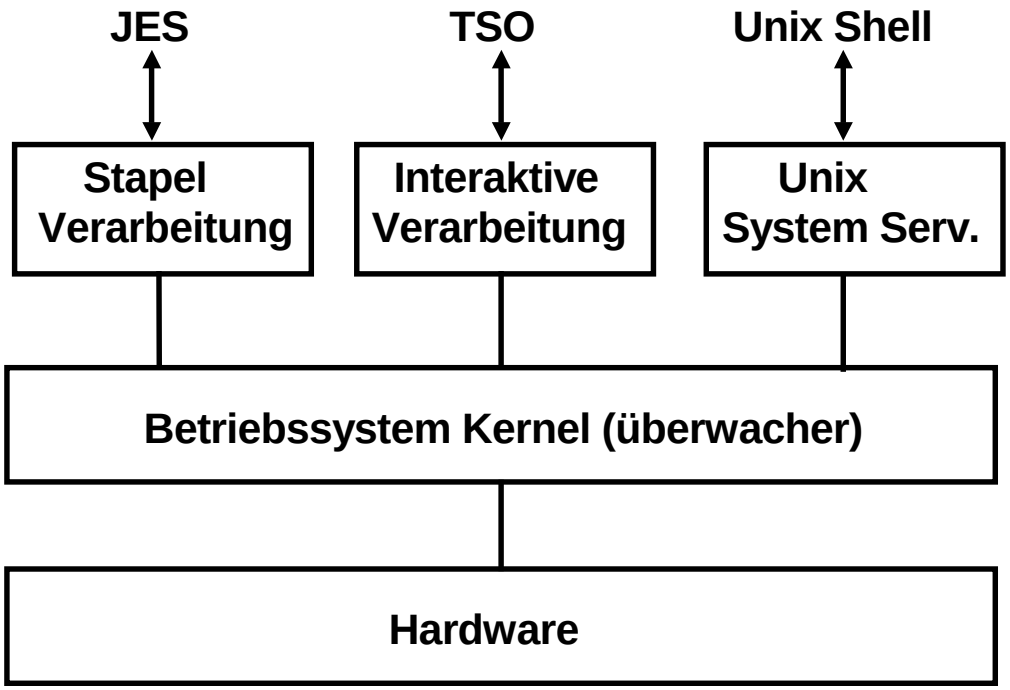


Windows NT Überwacher
Kernel Mode Funktionen



Windows NT Überwacher
Erweiterte Systemfunktionen

Virtual DOS Machine verarbeitet 16 Bit Anwendungen
Windows on Windows Subsystem verarbeitet 16 Bit Windows Apps.

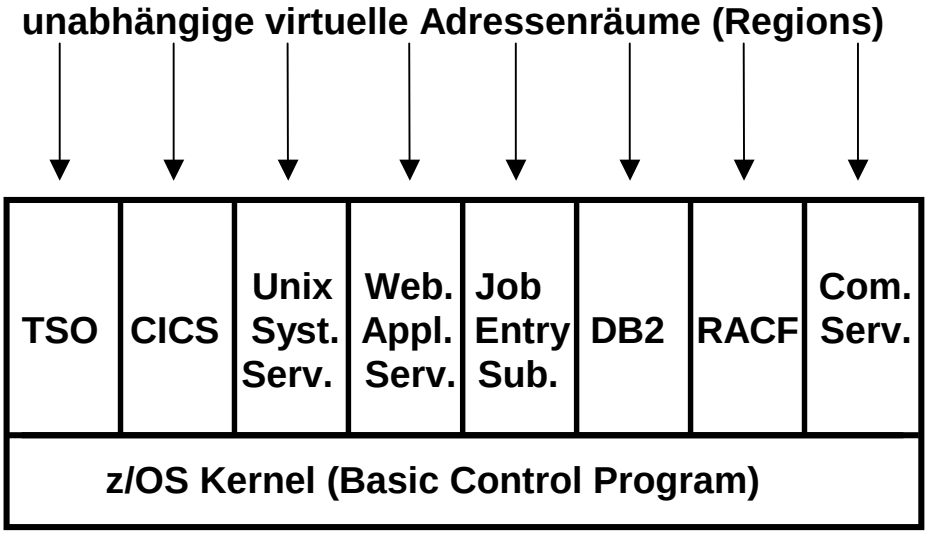


z/OS Grundstruktur

Die drei wichtigsten Subsysteme:

JES (Job Entry Subsystem) für die Stapelverarbeitung
TSO (Time Sharing Option) für die interaktive Verarbeitung (z.B. Programmentwicklung)
Unix System Services, Posix kompatibles Unix Subsystem

z/OS Grundstruktur



Der z/OS Kernel unterstützt eine Vielzahl von virtuellen Adressenräumen, die im z/OS Jargon als Regions bezeichnet werden.

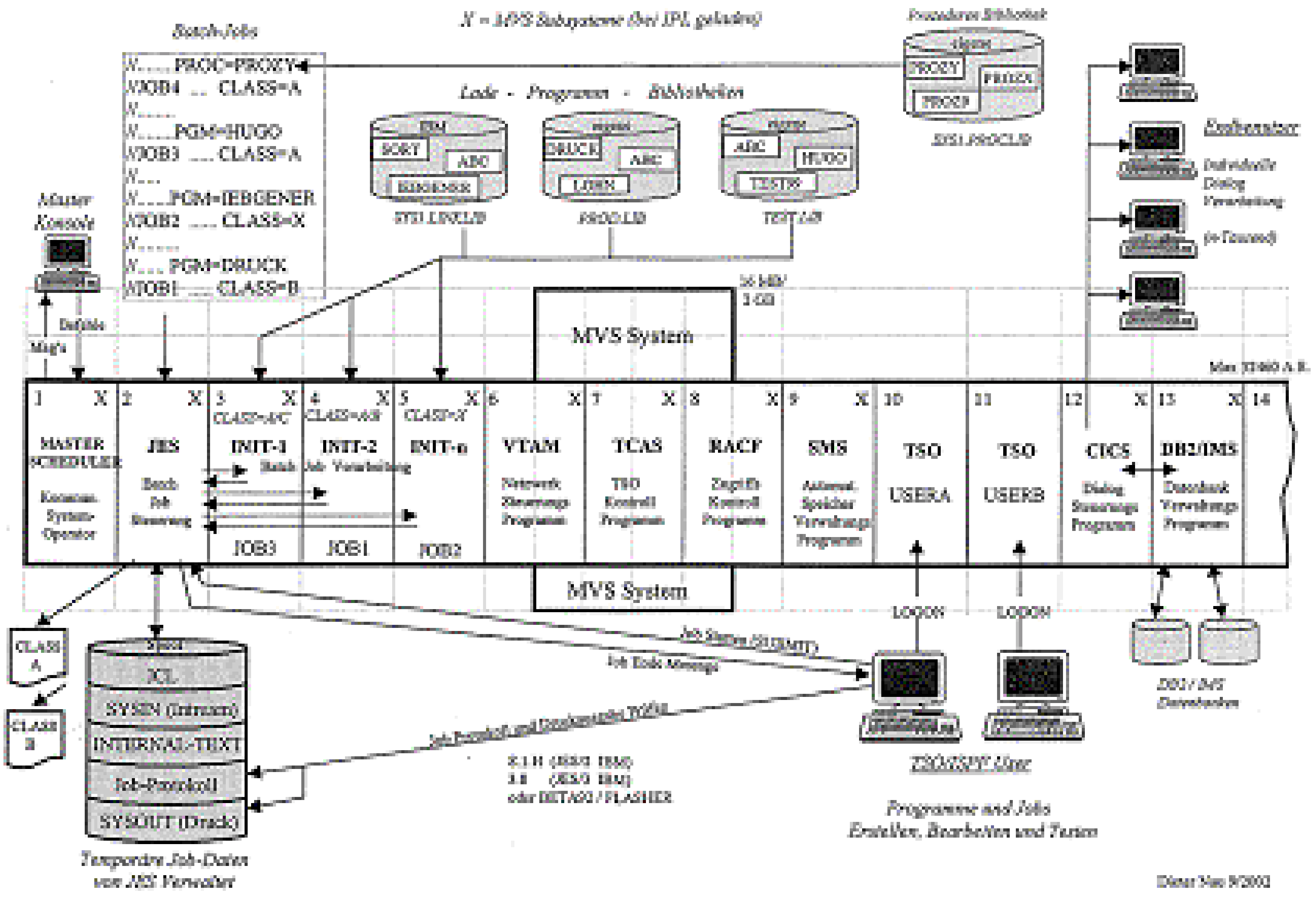
Einige der Regions beherbergen Subsysteme, die Teil des Betriebssystems sind, aber im Benutzerstatus laufen. Einige der (zahlreichen) Subsysteme sind:

- **CICS Transaktionsverarbeitung**
- **TSO Shell, Entwicklungsumgebung**
- **USS Unix kompatible Shell, Entwicklungsumgebung**
- **WAS WebSphere Web Application Server**
- **JES Job Entry Subsystem**
- **DB2 relationale Datenbank**
- **RACF Sicherheitssystem**
- **Communications Server**

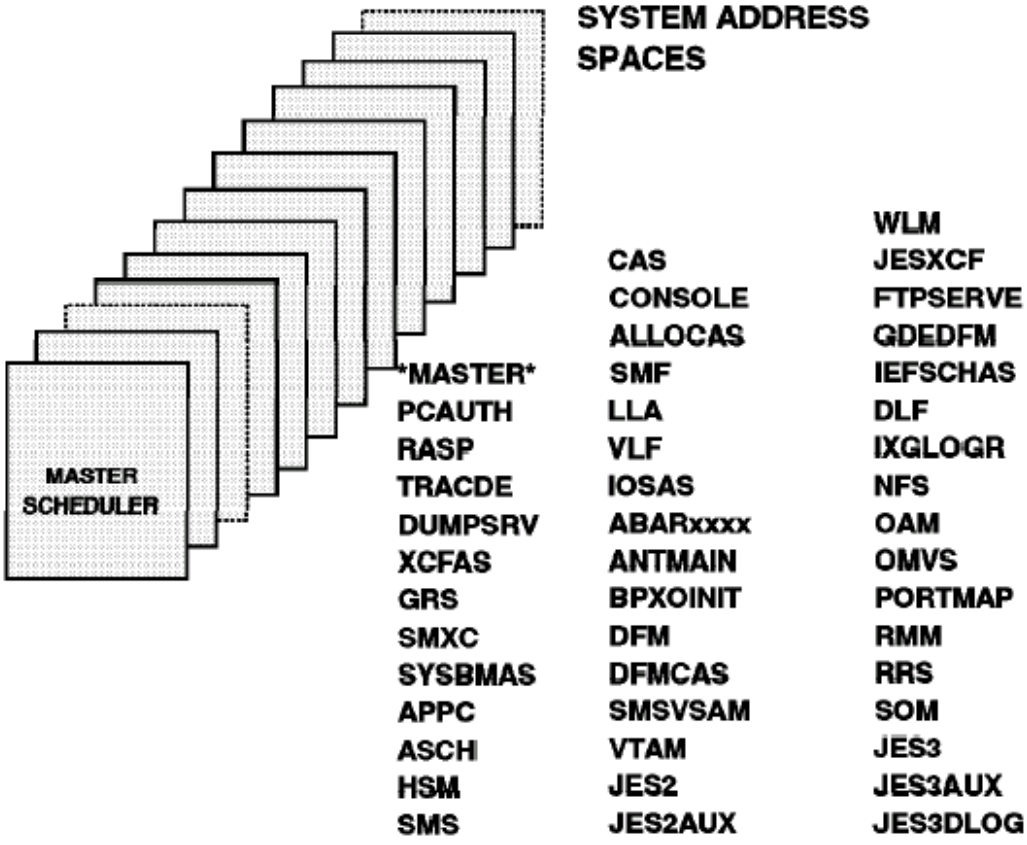
OS/390 Prozessverwaltung

Ein OS/390 Prozess besteht aus mehreren Arbeitseinheiten, die als „Tasks“ bezeichnet werden. Eine „Main Task“ generiert Subtasks und entspricht in etwa einem Thread in Unix oder NT. Wenn ein Programm gestartet wird, erstellt OS/390 hierfür einen Main Task Control Block (TCB). Das Programm kann weitere Subtasks erstellen mit Hilfe des ATTACH System Aufrufes.

Da der ATTACH Overhead relativ groß ist, implementieren zeitkritische Subsysteme wie z.B. CICS ihr eigenes Subtasking. Im Überwachermodus existieren zusätzlich SMB’s und SRB’s. Windows kennt „Fibers“ (light weight threads, die durch die Anwendung gescheduled werden). Die CICS Portierung auf Windows benutzt Fibers.


X = MVS Subsysteme (bei IPL geladen)
Batch-Jobs
//JOB4 ... CLASS=A
//......PGM=HUGO
//JOB3 .... CLASS=A
//......PGM=IEBGENER
//JOB2 .... CLASS=X
//..... PGM=DRUCK
//JOB1 .... CLASS=B
Master Konsole
Lade - Programm - Bibliotheken
SORT
ABC
DRUCK
LOHN
HUGO
SYS1.LINKLIB
PROCLIB
SYS1.PROCLIB
PROXY
MVS System
MASTER SCHEDULER
JES
INIT-1
INIT-2
INIT-n
VTAM
TCAS
RACF
SMS
TSO
USERA
USERB
CICS
DB2/IMS
JOB3
JOB1
JOB2
LOGON
CLASS A
CLASS B
JCL
SYSIN (Intrusen)
INTERNAL-TEXT
Job-Protokoll
SYSOUT (Druck)
Temporäre Job-Daten von JES Verwaltet
Programme und Jobs
Erstellen, Bearbeiten und Testen
Endbenutzer
DB2 / IMS Datenbanken

System Address Spaces



Ähnlich wie allen anderen modernen Betriebssysteme starten OS/390 und z/OS eine ganze Reihe von Systemprozessen, die in eigenen virtuellen Adressenräumen (Regions) laufen, getrennt von den virtuellen Adressenräumen, die für Benutzerprozesse angelegt werden. Die Adressenräume der Systemprozesse sind durch Protektion Keys gegen einen unauthorisierten Zugriff geschützt.

Programmarten

Subsysteme sind Programmprodukte wie Datenbanken und Transaktionsmonitore, die Laufzeitumgebungen für eigentliche Benutzerprogramme zur Verfügung stellen.

Benutzerprogramme können sein:

- **klassische z/OS (bzw. OS/390) Hintergrundprogramme**
- **Kundenanwendungen, die unter der Kontrolle von CICS, IMS oder Websphere ablaufen, oder**
- **UNIX-Programme, die die UNIX System Services unter z/OS ausnutzen.**

Systems Management Funktionen werden für die Steuerung und Überwachung des Ablaufes benötigt. Es gibt sehr viele solcher Funktionen von der IBM und Drittanbietern zur Überwachung des Betriebssystems und, da sich das Betriebssystem in weiten Bereichen selbst steuert, zur Überwachung der Middleware und der Kundenanwendungen.

jedi.informatik.uni-leipzig.de

139.18.4.35:80

S/390 Rechner an den Universität Leipzig

- **OS/390 V 2.7**
- **CICS**
- **IMS**
- **VSAM**
- **DB2**
- **Unix System Services**
- **JES**
- **Cobol**
- **Fortran**
- **C++**
- **Java**
- **WebSphere**
- **S/390-Linux**

Studenten loggen sich mit PC oder Unix Clienten in den S/390 Server ein.

Zugang mit einem gängigen 3270 Emulator, z.B. als Applet in einem Web Browser implementiert.

Verbindungsprotokoll : TCP/IP und Telnet

Direkter Web Zugriff ist möglich

Stapelverarbeitung vs. Interaktive Verarbeitung
Batch Processing vs. Interactive Processing

Eine interaktiven Client/Server Verarbeitung erfolgt synchron. Der Klient ruft ein Programm des Servers auf, und wartet (blokiert), bis die Ergebnisse des Servers zurückkommen.

Eine Stapelverarbeitung erfolgt asynchron. Ein Klient ruft ein Server Programm auf, und wendet sich dann anderen Aufgaben zu. Der Klient fragt nicht, wann der Server mit der Stapelverarbeitung fertig ist.

Die Benutzung eines Editors ist eine typische interaktive Anwendung. Die monatliche Gehaltsabrechnung in einem Großunternehmen ist typisch für eine Stapelverarbeitung.

Eine Stapelverarbeitung erfolgt häufig in mehreren Schritten. Ein Beispiel sind die Buchungsvorgänge für die monatliche Kreditabrechnung in einer Bank. Diese könnte z.B. aus den folgenden Schritten bestehen:

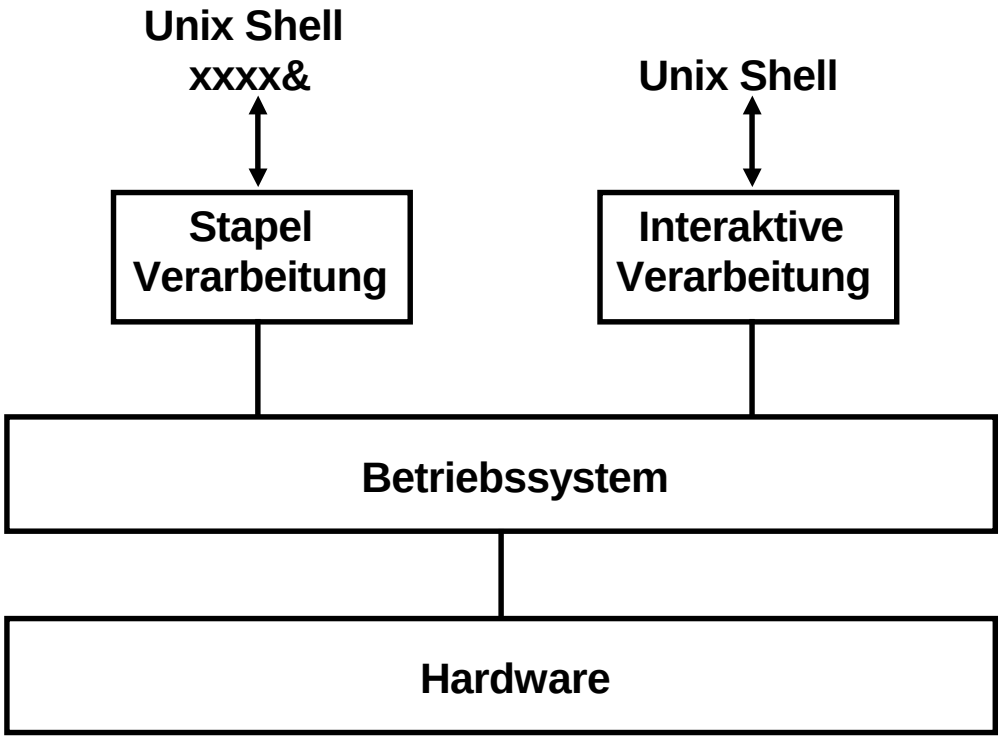
1. **Darlehnskonto abrechnen, Saldo um Tilgungsrate verändern**
2. **Tilgung und Zinsen im laufenden Konto (Kontokorrent) auf der Sollseite buchen**
3. **Globales Limit überprüfen**
4. **Bilanzpositionen (Konten)**
5. **G+V Positionen (Gewinn- und Verlust Konten)**
6. **Zinsabgrenzung monatlich für jährliche Zinszahlung**
7. **Bankmeldewesen (ein Kunde nimmt je 90 000.- DM bei 10 Banken auf, läuft am Stichtag)**

Ein derartiger Vorgang wird als „Job“ bezeichnet. Ein Job besteht aus einzelnen „Job Steps“.

Ein Stapelverarbeitungsauftrag interagiert während seiner Ausführung nicht mit dem Benutzer.

(Er kann während der Ausführung temporär unterbrochen und wieder aufgenommen werden, jenachdem ob Aufträge mit höherer Priorität die Ressourcen dringender benötigen).

Bei einer interaktiven Verarbeitung monitored der Benutzer die Ausführung und wartet auf die Ergebnisse.



Unix Grundstruktur

Die Stapelverarbeitung ist ein Sonderfall der interaktiven Verarbeitung. In der Shellsprache werden Batch-Aufträge durch ein nachgestelltes „&“ gekennzeichnet.

Batch Processing
Stapelverarbeitung

Batch-Jobs, auch Stapelverarbeitung oder Batch-Prozesse genannt, sind Programmroutinen, die zeit- oder ereignisgesteuert ohne Interaktion mit dem Benutzer ablaufen.

Die Hintergrundprogramme lesen Daten ein und liefern Rückgabewerte sowie Statusinformationen wie Laufzeit, Störungen und Fehlerbeschreibungen zurück. Ein Job-Scheduler startet und überwacht die Batch-Läufe. Praktisch alle Computersysteme verfügen über solche Mechanismen.

Selbst Windows-Desktops verwenden Jobs, beispielsweise zur Datensicherung oder zum Ausdrucken von Dokumenten.

Firmen verwenden die Batch-Steuerung unter anderem dazu, Massendaten wie Lohnabrechnungen zu verarbeiten, Vertriebszahlen auszuwerten oder Geschäftsinformationen in ein Data Warehouse zu laden.

Submission eines Jobs

Ein „Job Control Programm“ (Äquivalent einer Bat File unter DOS 6.22) besteht aus einer Reihe von prozeduralen Befehlen , und wird in der „Job Control Language“ (JCL) erstellt.

Die JCL hat ihren Ursprung im Lochkartenzeitalter. Jeder JCL Befehl beginnt mit den beiden Zeichen „ // „ und hat keinen Delimiter (z.B. „ ; „ in C++), sondern statt dessen eine feste Befehlslänge von genau 80 Zeichen (genau genommen 72 Zeichen plus 8 folgende Steuerzeichen).

Beispiel für einen JCL Befehl:

```
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
```

RECFM, FB, LRECL und BLKSIZE sind Schlüsselwörter der JCL Sprache.

Der Befehl besagt, daß die hiermit angesprochene Datei (bzw. ihr Data Control Block, DCB) ein Fixed Block (FB) Record Format (RECFM) hat (alle Datensätze haben die gleiche Länge), dessen Länge (Logical Record Length LRECL) 80 Bytes beträgt, und daß für die Übertragung vom/zum Hauptspeicher jeweils 5 Datensätze zu einem Block von (Blocksize BLKSIZE) 400 Bytes zusammengefaßt werden.

Literatur

M.Winkler: „MVS/ESA JCL“. Oldenbourg, 3. Auflage, 1999

Beispiel eines DOS 6.22 Jobs

auftrag.bat

```
f:
cd programme
able
baker
charlie
cd \
c:
```

f:\programme

able.exe	xxx.dat
baker.exe	yyy.dat
charlie.exe	zzz.dat
	aaa.dat

able

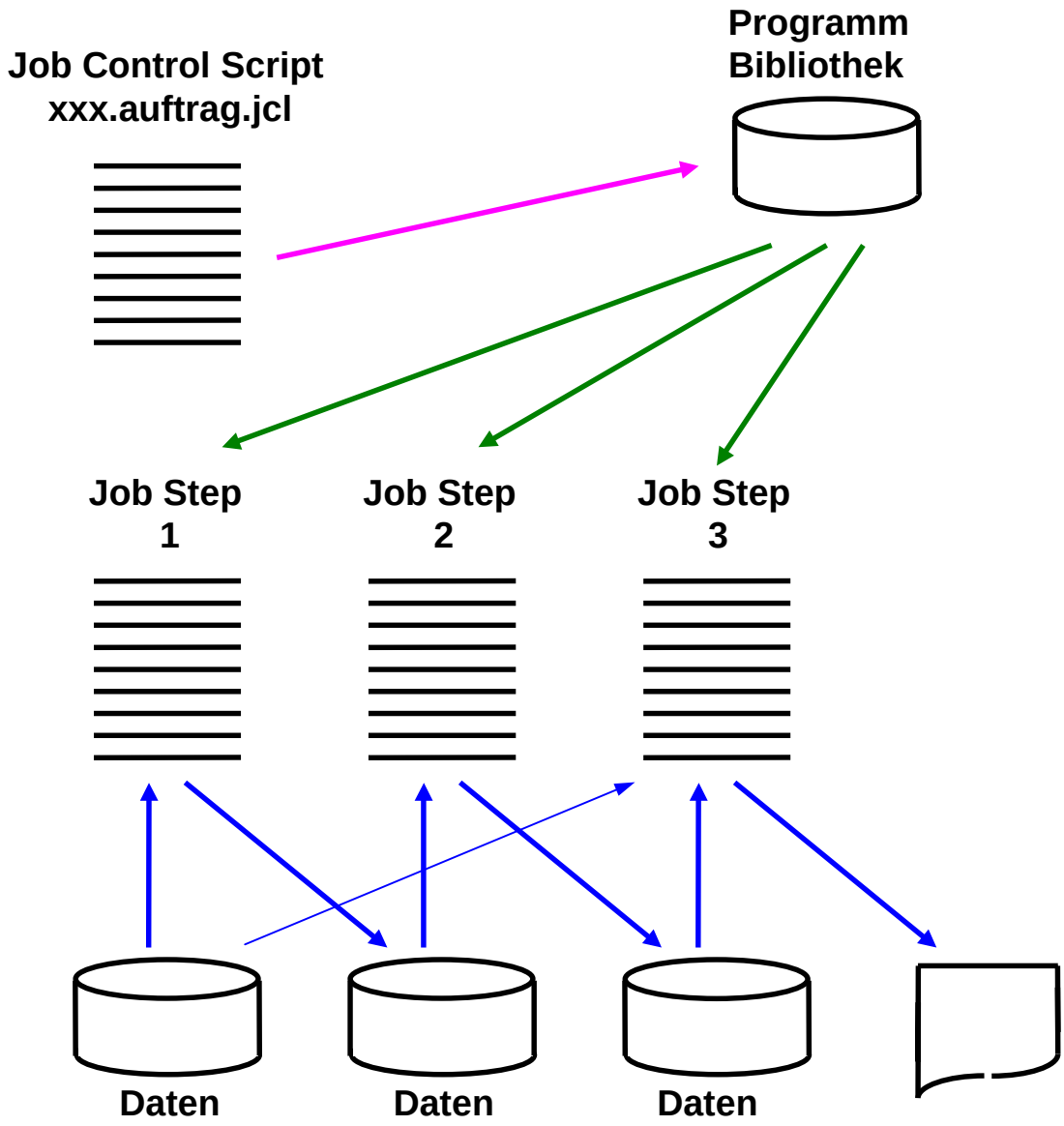
```
read xxx.dat
write yyy.dat
end
```

baker

```
read yyy.dat
write zzz.dat
end
```

charlie

```
read zzz.dat
read xxx.dat
write aaa.dat
end
```



Konzept eines OS/390 Jobs

Da das JCL Programm die verwendeten Dateien angibt, ist ein „late Binding“ der verwendeten Dateinen an die auszuführenden Programme möglich.

Eine „Cataloged Procedure“ ist ein JCL Programm, welches vom Benutzer für eine spätere Verwendung zwischengespeichert wird (.z.B. in einer vom Benutzer erstellten Library JCLLIB) und bei Bedarf mittels eines JCL Befehls aufgerufen wird.

Submission eines Jobs

Die Ablaufsteuerung für die einzelnen Job Steps innerhalb eines Jobs erfolgt durch ein Job Script, unter OS/390 als „Job Control Programm“ bezeichnet.

Ein Job Script ist in einer Script Sprache geschrieben. Unter Unix sind dies häufig Perl oder Tcl/Tk; unter Windows sind es z.B. VBScript, JavaScript oder Rexx. Unter OS/390 sind es meistens Rexx oder JCL.

Rexx ist auch außerhalb S/390 weit verbreitet und auf zahlreichen Plattformaen verfügbar; „it is just another Script Language“. Dagegen ist JCL (Job Control Language) nur unter OS/390 verfügbar, sehr ungewöhnlich, trotzdem aber sehr populär.

JCL hat ihren Ursprung im Lochkartenzeitalter. Jeder JCL Befehl beginnt mit den beiden Zeichen „ // „ und hat keinen Delimiter (z.B. „ ; „ in C++) sondern statt dessen eine feste Befehlslänge von genau 80 Zeichen.

Beispiel für einen JCL Befehl:

```
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
```

RECFM, FB, LRECL und BLKSIZE sind Schlüsselwörter der JCL Sprache.

Der Befehl besagt, daß die hiermit angesprochene Datei (bzw. ihr Data Control Block, DCB) ein Fixed Block (FB) Record Format (RECFM) hat (alle Datensätze haben die gleiche Länge), diese Länge (Logical Record Length LRECL) 80 Bytes beträgt, und daß für die Übertragung vom/zum Hauptspeicher jeweils 5 Datensätze zu einem Block von (Blocksize BLKSIZE) 400 Bytes zusammengefaßt werden.

Literatur: M.Winkler: „MVS/ESA JCL“. Oldenbourg, 3. Auflage, 1999

Voraussetzung für die Ausführung eines Jobs ist, daß benutzte Programme und Dateien (Data Sets, Files) bereits existieren.

Erstellung und Eingabe (Submission) eines Jobs erfordert die folgenden Schritte:

1. **Zuordnung einer Datei, welche das Job Control Programm enthalten soll. (Kann unter ISPF geschehen).**

2. **Editieren und Abspeichern der JCL Datei (unter ISPF oder TSO)**

3. **Submission**

Da das JCL Programm die verwendeten Dateien angibt, ist ein „late Binding“ der verwendeten Dateinen an die auszuführenden Programme möglich.

Eine „Cataloged Procedure“ ist ein JCL Programm, welches vom Benutzer für eine spätere Verwendung zwischengespeichert wird (.z.B. in einer vom Benutzer erstellten Library JCLLIB) und bei Bedarf mittels eines JCL Befehls aufgerufen wird.

Die wichtigsten Sprachen

- **Cobol**
- **PL/1**
- **(Assembler)**
- **Fortran**
- **C/C++**
- **Java**

Skripte

- **Shellskripte unter Unix**
- **BAT-Files unter DOS***
- **JCL (z/OS Job Control Language)**

Skriptsprachen

- **TCL/TK**
- **Pearl**
- **PHP**
- **REXX**

REXX

Interpretative Scriptsprache, verfügbar für die meisten Betriebssysteme, vergleichbar zu Pearl oder Tcl/tk. . Weit verbreitet bei den Benutzern von z/OS Rechnern. Vollwertige Programmiersprache.

REXX Programmierbeispiel

```
/* A short program to greet you                          */
/* First display a prompt                                */

say ‘Please type your name and then press ENTER:‘
parse pull answer
                      /* Get the reply into answer   */

/* If nothing was typed, then use a fixed greeting   */
/* otherwise echo the name politely                  */

if answer=‘‘   then say ‘Hello Stranger! ‘
              else say ‘ Hello ‘ answer ‘!‘
```

```
//SPRUTHC  JOB (123456),'SPRUTH',CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),
//             NOTIFY=&SYSUID,TIME=1440
//PROCLIB  JCLLIB ORDER=CBC.SCBCPRC
//CCL      EXEC PROC=EDCCB,
//           INFILE='SPRUTH.TEST.C(HELLO1)'
//           OUTFILE='SPRUTH.TEST.LOAD(HELLO1),DISP=SHR'
```

Ein einfaches JCL Script

JOB Statement markiert den Anfang eines Jobs

EXEC Statement bezeichnet Prozedur, die ausgeführt werden soll

PROC Statement gibt den Anfang einer Procedur an

DD Statement bezeichnet die zu benutzenden Dateien (hier nicht verwendet)

/* Statement markiert das Ende von Daten, die in die JCL Statements eingeführt werden

Druckwerke Reichenbach 1/1

Batch Processing
Stapelverarbeitung

Batch-Jobs, auch Stapelverarbeitung oder Batch-Prozesse genannt, sind Programmroutinen, die zeit- oder ereignisgesteuert ohne Interaktion mit dem Benutzer ablaufen.

Die Hintergrundprogramme lesen Daten ein und liefern Rückgabewerte sowie Statusinformationen wie Laufzeit, Störungen und Fehlerbeschreibungen zurück. Ein Job-Scheduler startet und überwacht die Batch-Läufe. Praktisch alle Computersysteme verfügen über solche Mechanismen.

Selbst Windows-Desktops verwenden Jobs, beispielsweise zur Datensicherung oder zum Ausdrucken von Dokumenten.

Firmen verwenden die Batch-Steuerung unter anderem dazu, Massendaten wie Lohnabrechnungen zu verarbeiten, Vertriebszahlen auszuwerten oder Geschäftsinformationen in ein Data Warehouse zu laden.

Job Scheduler
Job Entry Subsystem

Job-Scheduler arbeiten nach dem Master/Agent-Konzept.

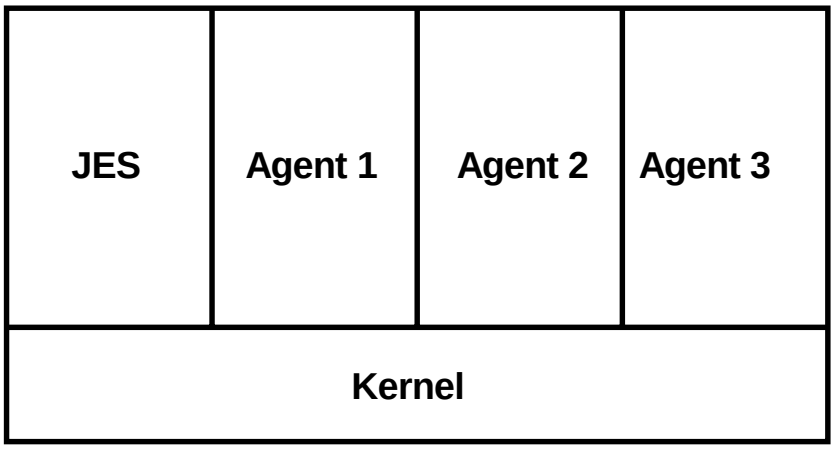
Der Master (Job entry subsystem, entweder JES2 oderJES3)

- **initiiert die Batches,**
- **reiht sie je nach Priorität in Warteschlangen ein und**
- **kontrolliert deren Ablauf.**

Die Agenten sind selbständige Prozesse, die in eigenen virtuellen Adressräumen laufen. Sie

- **führen die Jobs aus und**
- **senden Statusinformationen an die Steuerungskomponente und/oder**
- **fertigen Laufzeitberichte an.**

/



Job Entry Subsystem (JES)

Auf einem Großrechner laufen in der Regel zahlreiche Jobs gleichzeitig und parallel (hunderte oder tausende von z. T. langlaufenden Jobs während eines Tages).

Die Übergabe eines neuen Jobs an das Betriebssystem, das Queuing von Jobs, die Prioritäts- und Ablaufsteuerung zahlreicher Jobs untereinander, Startzeit und Wiederholfrequenz, Durchsatz-optimierung sowie die Zuordnung oder Sperrung von Resourcen ist die Aufgabe eines Job Steuerungssystems.

Unter Windows ist dies der „Windows Scripting Host“. Unter Unix existieren verschiedene Software Produkte, z.B. S+ batch, fbatch und viele andere.

Die Übergabe (Submission) eines Stapelverarbeitungs-Jobs an das Betriebssystem erfolgt durch einen Benutzer; bei einem Großrechner durch einen spezifisch hierfür abgestellten Bediener (Operator). Unter z/OS wird die Job Submission mit Hilfe des Job Entry Subsystems (JES) automatisiert.

JES bewirkt:

- **Zu Anfang der Bearbeitung die Zuordnung von Eingabe und Ausgabe Ressourcen**
- **Während der Verarbeitung die Zuordnung von Ressourcen wie CPU´s und Hauptspeicher**
- **Nach Abschluß der Bearbeitung die Freigabe der Ressourcen. Sie werden damit für andere Jobs verfügbar.**

Jobs werden an das Betriebssystem über JES gesteuerte Queue Server (Initiator) übergeben. Jeder Initiator bedient einen einzigen virtuellen Adressenraum, in dem er auch selbst untergebracht ist..

Wenn ein Job abgeschlossen wird, wird der Initiator des Adressraums aktiv und JES sendet den nächsten Job, der darauf wartet, verarbeitet zu werden.

Server Zugriff

Unterschied zwischen Einzelplatzrechner und Client/Server Betriebssystemen.

Windows, Linux und und Unix werden für beides eingesetzt. OS/390 und z/OS sind reinrassige Server Betriebssysteme. Andere Beispiele für Server Betriebssysteme: Tandem NonStop, DEC OpenVMS.

Ein Server Zugriff benötigt spezielle Client Software. Drei Client Alternativen:

1. **Selbstgeschriebene Anwendungen:**

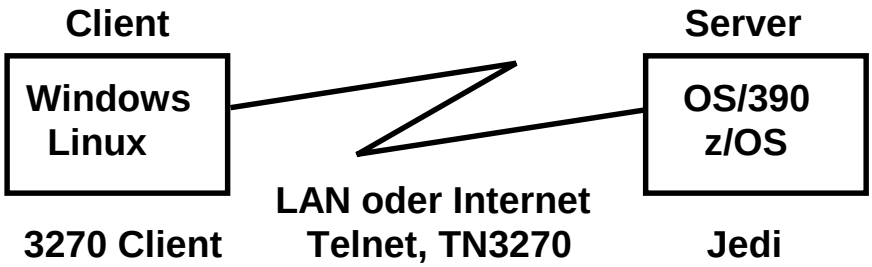
   Sockets, RPC, Corba, DCOM, RMI

2. **Zeilenorientierte Klienten:**

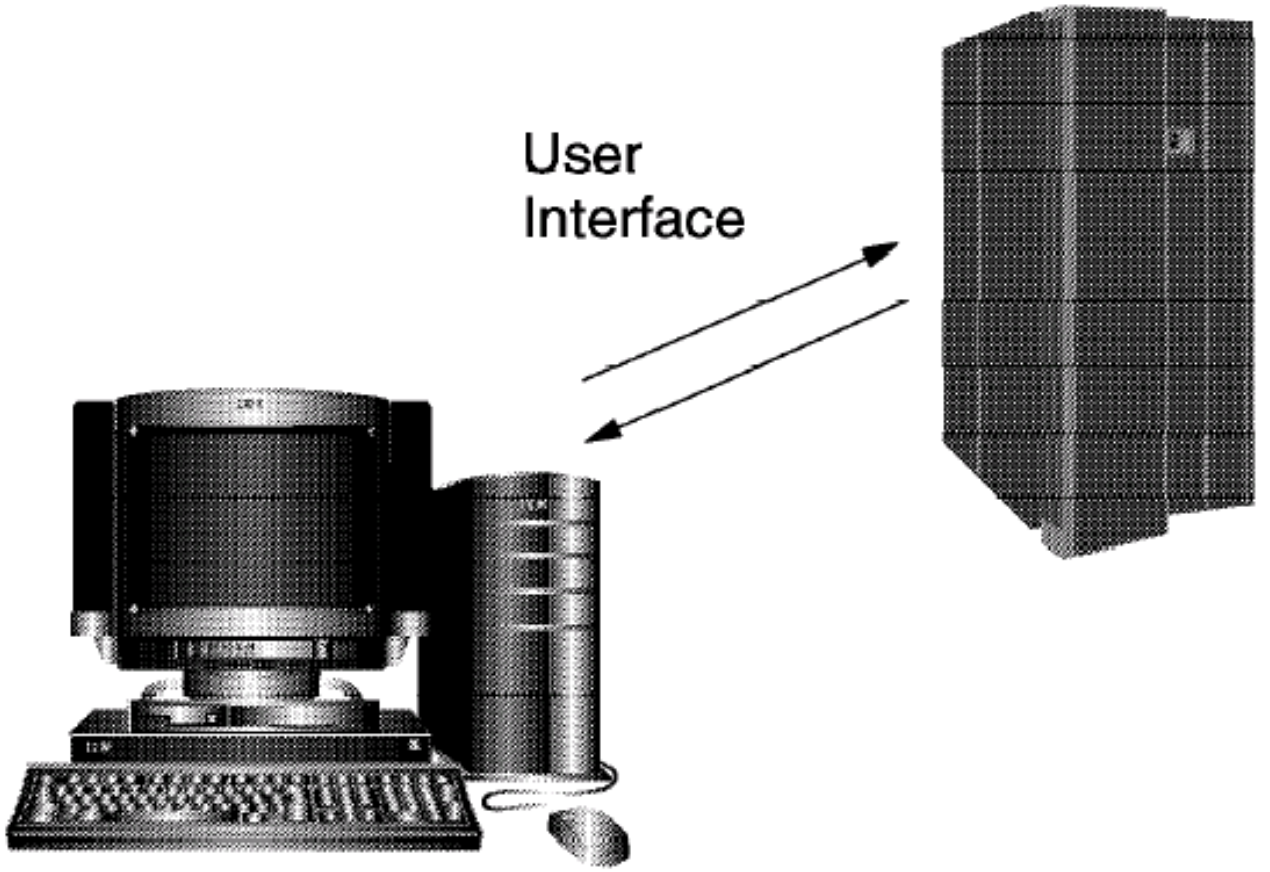
Unix Server	**Telnet Client, FTP**
OS/390 Server	**3270 Client (3270 Emulator)**
VMS Server	**VT 100 Client**

3. **Klienten mit graphischer Oberfläche:**

Windows Server	**Citrix Client**
WWW Server	**Browser Client**
SAP R/3 Server	**SAPGUI Client**
Unix Server	**XWindows, Motif**
z/OS und OS/390 Server	**Servlet, JSP Client**



TSO/E
Time Sharing Option/Extended



TSO ist eine zeilenorientierte Shell, vergleichbar mit der Windows „ „DOS Eingabeaufforderung“ oder den Unix/Linux Bourne, Korn, Bash oder C-Shells.

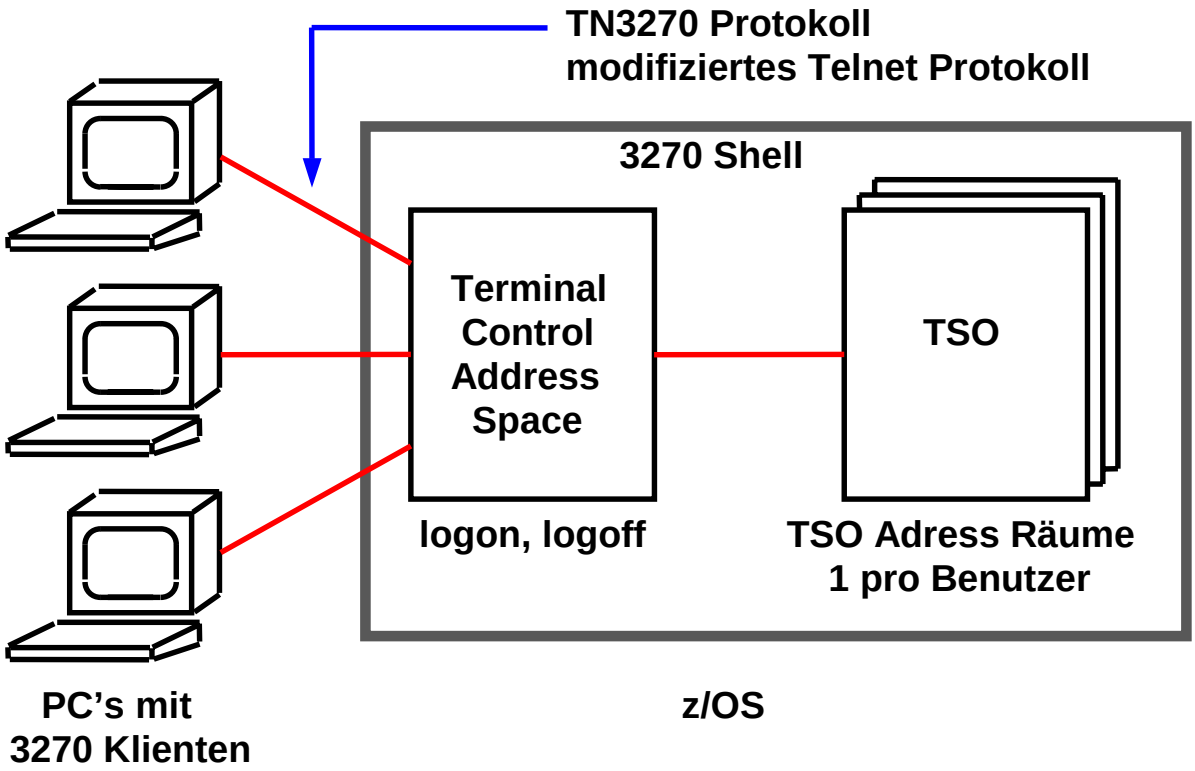
TSO („Time Sharing Option“)

Vergleichbar mit dem ähnlichen UNIX Time Sharing. Hauptsächliche Anwendung: Software Entwicklung und Test, System Administration

Alternative für die Software Entwicklung: Nutzung des VM Betriebssystems in einer separaten LPAR

Systemprogrammierer verwenden TSO um Files zu editieren, Steuerungen vorzunehmen, Systemparameter zu setzen und einen Job Status zu überprüfen.

„Full Screen“ und Command Level Schnittstelle verfügbar. Die Full Screen Komponente wird als ISPF - Interactive System Productivity Facility bezeichnet.



Der TSO Terminal Control Prozess (TCAS) arbeitet in einem eigenen Adressenraum. Er nimmt Nachrichten von den einzelnen TSO Klienten entgegen und leitet sie an den für den Benutzer eingerichteten separaten TSO Adressenraum weiter.

```
TCPIP MSG10 ==> SOURCE DATA SET = SYS1.LOCAL.VTAMLST(USSTCPIP)

 08/01/01                      W E L C O ME  T O                          17:17:08


                   SSSSSS    //   3333333   9999999   0000000
                  SS         //   33   33   99   99   00   00
                 SS          //        33   99   99   00   00
                  SSSS       //     33333   9999999   00   00
                     SS      //        33         99   00   00
                    SS      //   33   33   99   99   00   00
                SSSSSSS     //   3333333   9999999   0000000


 YOUR TERMINAL NAME IS : SC0TCP05          YOUR IP ADDRESS IS : 217.081.157.057


                          APPLICATION DEVELOPMENT SYSTEM
                               OS/390 RELEASE 2.7.0

 ===> ENTER "L " FOLLOWED BY THE APPLID YOU WISH TO LOGON TO.  EXAMPLE "L TSO"
      FOR TSO/E OR "L C001" FOR THE CICSC001 CICS APPLICATION.


 _
```

```
CUSTOMPAC MASTER APPLICATION MENU
 OPTION ===>                                          SCROLL ===> PAGE

   IS  ISMF     - Interactive Storage Management Facility
   P   PDF      - ISPF/Program Development Facility
   ATC ATC      - Application Testing Collection
   ART ARTT     - Automated Regression Testing Tool
   DB2 DB2      - Perform DATABASE 2 interactive functions
   QMF QMF      - QMF Query Management Facility
   C   CPSM     - CICSPlex/SM
   M   MQ       - MQSeries
   IP  IPCS     - Interactive Problem Control Facility
   OS  SUPPORT  - OS/390 ISPF System Support Options
   OU  USER     - OS/390 ISPF User Options
   SM  SMP/E    - SMP/E Dialogs
   SD  SDSF     - System Display and Search Facility
   R   RACF     - Resource Access Control Facility
   DI  DITTO    - Data Interfile Transfer, Testing and Operations
   HC  HCD      - Hardware Configuration Definition
   S   SORT     - DF/SORT Dialogs
   BMR BMR READ - BookManager Read (Read Online Documentation)

  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

```
Menu  Utilities  Compilers  Options  Status  Help
 sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
                           ISPF Primary Option Menu

 0  Settings      Terminal and user parameters           User ID . : SPRUTH
 1  View          Display source data or listings        Time. . . : 18:20
 2  Edit          Create or change source data           Terminal. : 3278
 3  Utilities     Perform utility functions              Screen. . : 1
 4  Foreground    Interactive language processing        Language. : ENGLISH
 5  Batch         Submit job for language processing     Appl ID . : PDF
 6  Command       Enter TSO or Workstation commands      TSO logon : IKJACCNT
 7  Dialog Test   Perform dialog testing                 TSO prefix: SPRUTH
 8  LM Facility   Library administrator functions        System ID : DAVI
 9  IBM Products  IBM program development products       MVS acct. : ACCT#
EsssssssssssssssssssssssssssssssssssssssssssssssssN r    Release . : ISPF 4.5
e Licensed Materials - Property of IBM         e
e 5647-A01 (C) Copyright IBM Corp. 1980, 1997. e
e All rights reserved.                         e
e US Government Users Restricted Rights -      e s
e Use, duplication or disclosure restricted    e
e by GSA ADP Schedule Contract with IBM Corp.  e
DsssssssssssssssssssssssssssssssssssssssssssssssssM
 Option ===>
  F1=Help      F3=Exit     F10=Actions  F12=Cancel
```

Detaillierte Screen by Screen Tutorials sind zu finden unter

http://jedi.informatik.uni-leipzig.de

Dateisystem

Ein Dateisystem verbirgt die Eigenschaften der physikalischen Datenträger (Plattenspeicher, CD, evtl. Bandlaufwerke) weitestgehend vor dem Amwendungsprogrammierer.

Unix, NT

Dateien sind strukturlose Zeichenketten, welche über Namen identifiziert werden. Hierfür dienen Dateiverzeichnisse, die selbst wie Dateien aussehen und behandelt werden können.

Dateien werden sequentiell gelesen. Ein Direktzugriff wird mit Hilfe von Funktionen programmiert, die gezielt auf ein bestimmtes Zeichen in der Datei vor- oder zurücksetzen.

z/OS, OS/390

Das Dateisystem (Filesystem) wird durch die Formattierung eines physikalischen Datenträgers definiert. Bei der Formattierung der Festplatte wird die Struktur der Datei festgelegt. Es gibt unterschiedliche Formattierungen für Dateien mit direktem Zugriff (DAM), sequentiellen Zugriff (SAM) oder index-sequentiellen Zugriff (VSAM).

Dateizugriffe benutzen an Stelle eines Dateiverzeichnisse „Kontrollblöcke“, welche die Datenbasis für unterschiedliche Betriebssystemfunktionen bilden.

Konzepte: Datenorganisation

► UNIX/NT

- Dateien sind strukturlose Zeichenketten
- Zugriffsmethode: READ(fileid, buffer, length)
  - offset, length

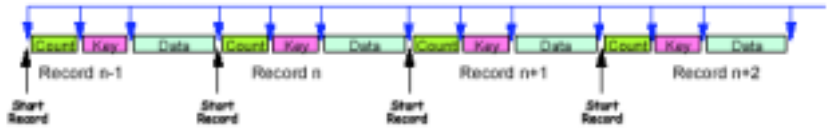
  - Fixed Block Architecture

- Zugriffsmethode: "raw"
  - Implementiert bei Datenbankanwendungen ausserhalb des Dateisystems

► S/390

- Record orientierter Zugriff
- Zugriffsmethode: GET recordid, buffer
  - Teil des Betriebssystems
  - sequentiel, indiziert, random
  - EXCP: Execute Channel Program: direct

- Count-Key-Data



- Dataset
  - Werden allokiert mit fester Grösse
  - Erweiterungen möglich über "Extends"
  - Können über mehrere Volumes gehen: "Spanned"

Data Sets und Files in z/OS und OS/390

Data Sets	Files
Access Methods	**UNIX Services**
z/OS data sets Record-oriented (F(B), V(B), U)	HFS files Byte-oriented
Two general methods VSAM (ESDS,KSDS,RRDS,LDS) Non-VSAM (SAM, PDS-E)	Hierarchical file structure directory/subdir/filename Path info max. 1023 char. File name max. 256 char.
Data set name (max. 44 chars.) UPPER CASE names	Mixed case, case sensitive names
Location/attr. stored in catalog	Stored in z/OS container data sets

z/OS and OS/390 verwenden zwei vollständig unterschiedliche Verfahren um Daten auf Plattenspeichern anzuordnen;

- **Traditional MVS Data sets and**
- **Unix files.**

DFSMS Klassen

Data Facility Storage Management System

Beim Neuanlegen einer Datei müssen angegeben werden:

Data Class

File System Attribute wie Record Format, Record Länge, Schlüssellänge bei VSAM Dateien

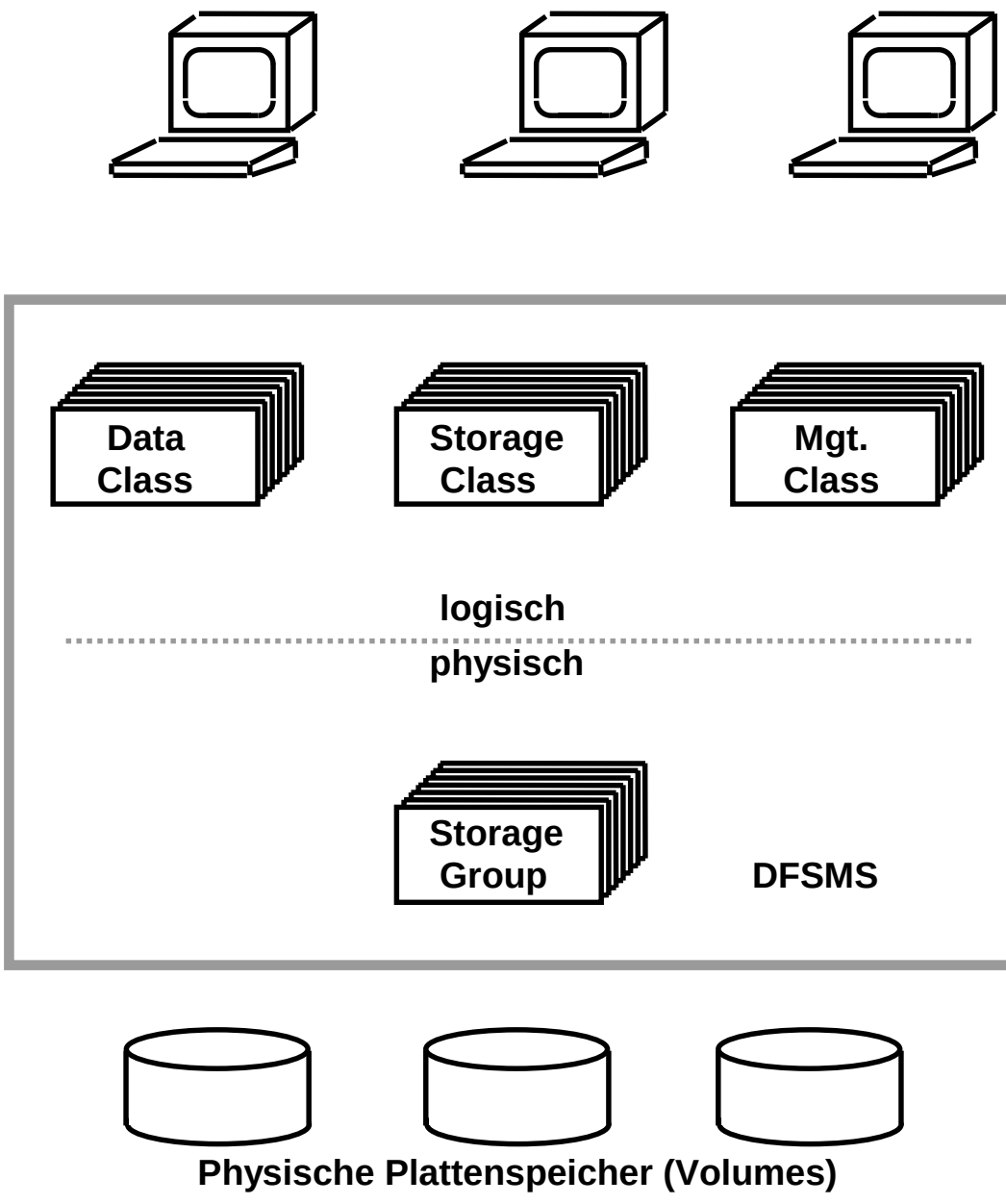
Storage Class

Performance Angaben wie Antwortzeit (ms), Nutzung von Read/Write Caching, Verwendung der Dual Copy Funktion

Management Class

Migration nach x Tagen, Anzahl Backup Versionen, schrittweiser Abbau der Backup Versionen, Löschen der Daten

DFSMS bewirkt die Zuordnung der Datei zu einer Storage Group, einer Gruppe von Plattenspeichern (Volumes)



Data Facility Storage Management System

Der Benutzer sieht nur die logische Sicht der Daten:

- **vorbereitete Datenmodelle in den Datenklassen**
- **in den Storageklassen festgelegte Serviceanforderungen**
- **Management Kriterien für die Auslagerung der Daten**

Lebenszyklus einer Datei
System managed Storage (SMS)

Anlegen der Datei durch den Benutzer

Benutzung (Schreiben und/oder Lesen der Daten)

Sicherungskopien anlegen

Platzverwaltung (freigeben / erweitern / komprimieren)

Auslagern von inaktiven Dateien, sowie Wiederbenutzung

Ausmustern von Sicherungskopien

Wiederherstellung von Dateien

Löschen der Datei

Benutzung eines z/OS Systems

Erste Aufgabe: Zuordung (allocate) von Dateien

Frage: Was soll das ? Habe ich unter Windows noch nie gemacht.

Falsch. Wenn Sie unter Windows eine neue Datei erstmalig anlegen, müssen Sie entscheiten, ob sie unter C: oder D: gespeichert wird.

Wenn Ihr Rechner über 24 Plattenspeicher verfügt,
z.B. C:, D:, E:, ...Z: ,
wird die Verwaltung schon etwas schwieriger.

Was machen Sie wenn Ihr z/OS Rechner über 60 000 Plattenspeicher verfügt ?

Sie verwenden für die Verwaltung eine z/OS Komponente System Managed Storage (SMS).

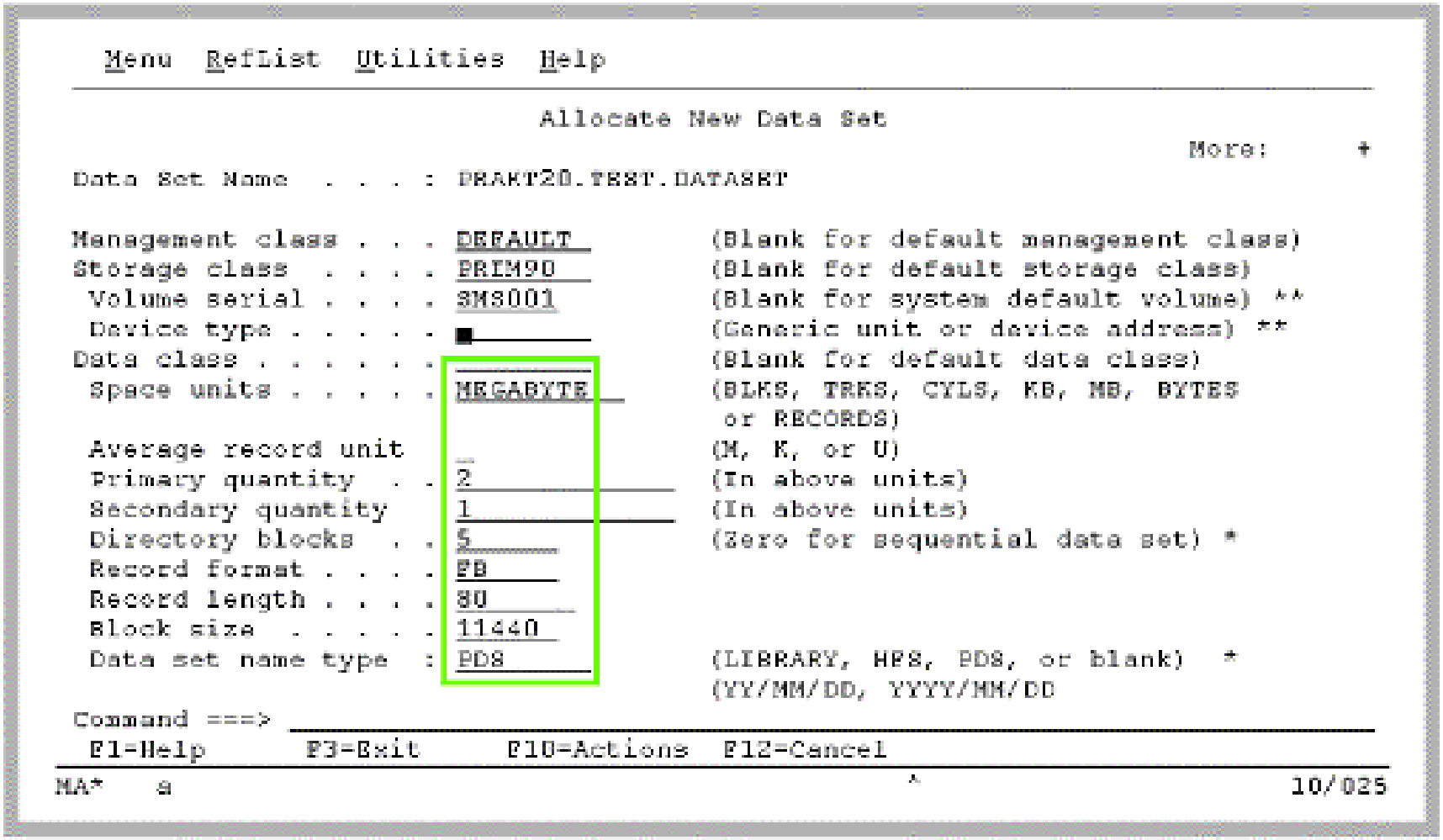
Wenn Sie Platz für eine neue Datei brauchen, melden Sie dies bei SMS an. Dieser Vorgang wird als Allocation bezeichnet.

Erstellen einer Datei: Entscheidungen

Benutzer, die Datenverarbeitung betreiben, haben beim Umgang mit den Daten täglich viele Entscheidungen zu treffen. Zusätzlich zum Umgang mit Themen, die nur die Daten oder die Anwendung betreffen, müssen sie die Speicherverwaltungsmaßnahmen der Installationen kennen. Sie müssen sich außerdem mit den Themen auseinandersetzen, die Format, Bearbeitung und Positionierung der Daten betreffen:

- **Welchen Wert soll die Blockungsgröße haben? Wieviel Speicherplatz ist erforderlich?**
- **Welcher Einheitentyp soll verwendet werden? Sollen die Daten in den Cache geschrieben werden? Soll eine Fehlerbehebung durchgeführt werden?**
- **Wie oft soll eine Sicherung oder Migration durchgeführt werden? Soll die Sicherung/Migration erhalten bleiben oder gelöscht werden?**
- **Welche Datenträger stehen für die Dateipositionierung zur Verfügung?**

Wenn die Verwendung von Datenverarbeitungsservices vereinfacht werden soll, müssen dem System einfachere Schnittstellen zur Verfügung gestellt werden. Insbesondere JCL ist einer der Bereiche, in denen Vereinfachungen vorgenommen werden.


Menu RefList Utilities Help
Allocate New Data Set
More: +
Data Set Name . . . : DRAKT20.TEST.DATASET
Management class . . . DEFAULT (Blank for default management class)
Storage class . . . . PRIM90 (Blank for default storage class)
Volume serial . . . . SMS001 (Blank for system default volume) **
Device type . . . . . (Generic unit or device address) **
Data class . . . . . (Blank for default data class)
Space units . . . . MEGABYTE (BLKS, TRKS, CYLS, KB, MB, BYTES or RECORDS)
Average record unit (M, K, or U)
Primary quantity . . 2 (In above units)
Secondary quantity 1 (In above units)
Directory blocks . . 5 (Zero for sequential data set) *
Record format . . . FB
Record length . . . 80
Block size 11440
Data set name type : PDS (LIBRARY, HFS, PDS, or blank) *
(YY/MM/DD, YYYY/MM/DD
Command ===>
F1=Help F3=Exit F10=Actions F12=Cancel
MA* a 10/025

DB2 relationale Datenbank

***DB2 Universal Database (UDB)*, ist das z/OS (objekt-) relationale Datenbank-Produkt.**

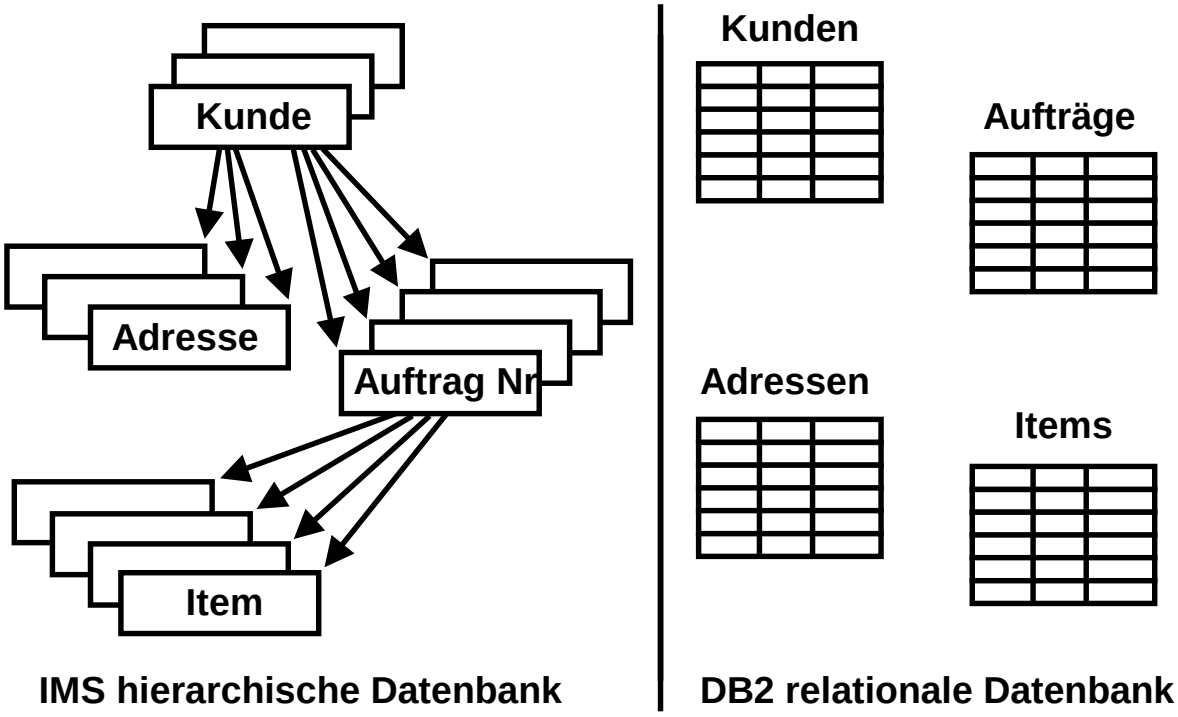
Identische Implementierung für alle UNIX-, Linux-, OS/2-, und Windows-Betriebssysteme.

Eine zweite getrennte Implementierung mit dem gleichen Namen und erweitertem Funktionsumfang ist für das z/OS-Betriebssystem verfügbar, Obwohl es sich um zwei getrennte Implementierungen handelt, ist die Kompatibilität sehr gut.

Neben Oracle- und Microsoft-SQL ist DB2 eines der drei führenden relationalen Datenbankprodukte. Unter z/OS ist es neben IMS die am häufigsten eingesetzte Datenbank. Andere populäre z/OS-Datenbanksysteme sind IDMS der Fa. Computer Associates, Oracle sowie Adabas der Fa. Software AG.

DB2 ist eine Server-Anwendung, die grundsätzlich in einem getrennten Adressraum läuft. Wie bei allen Server-Anwendungen ist ein Klient erforderlich, um auf einen DB2-Server zuzugreifen. Der Klient kann ein Anwendungsprogramm auf dem gleichen Rechner sein, oder auf einem getrennten Rechner laufen.

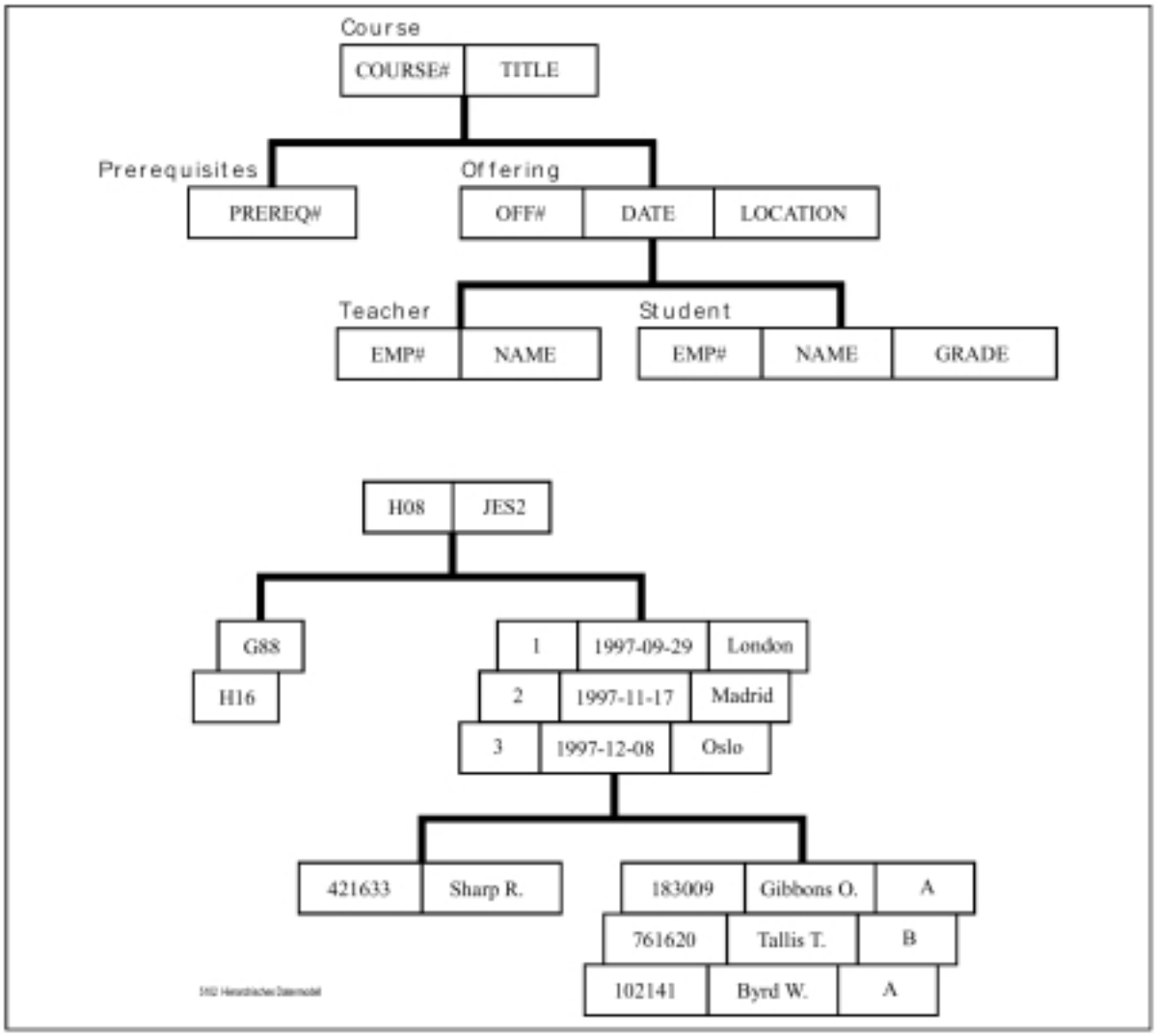
Der Zugriff kann mit Hilfe von SQL-Statements erfolgen, die in einem Anwendungsprogramm eingebettet sind. Alternativ existieren eine Reihe spezifischer SQL-Klienten-Anwendungen.



IMS Datenbanksystem

IMS ist im Gegensatz zu DB2 ein nicht-relationales, hierarchisches Datenbanksystem. IMS ermöglicht höhere Transaktionsraten als DB2.

Der CICS Transaktionsmonitor kann auf IMS Daten Zugreifen. Daneben verfügt IMS über einen eigenen Transaktionsmonitor IMS DB/DC.



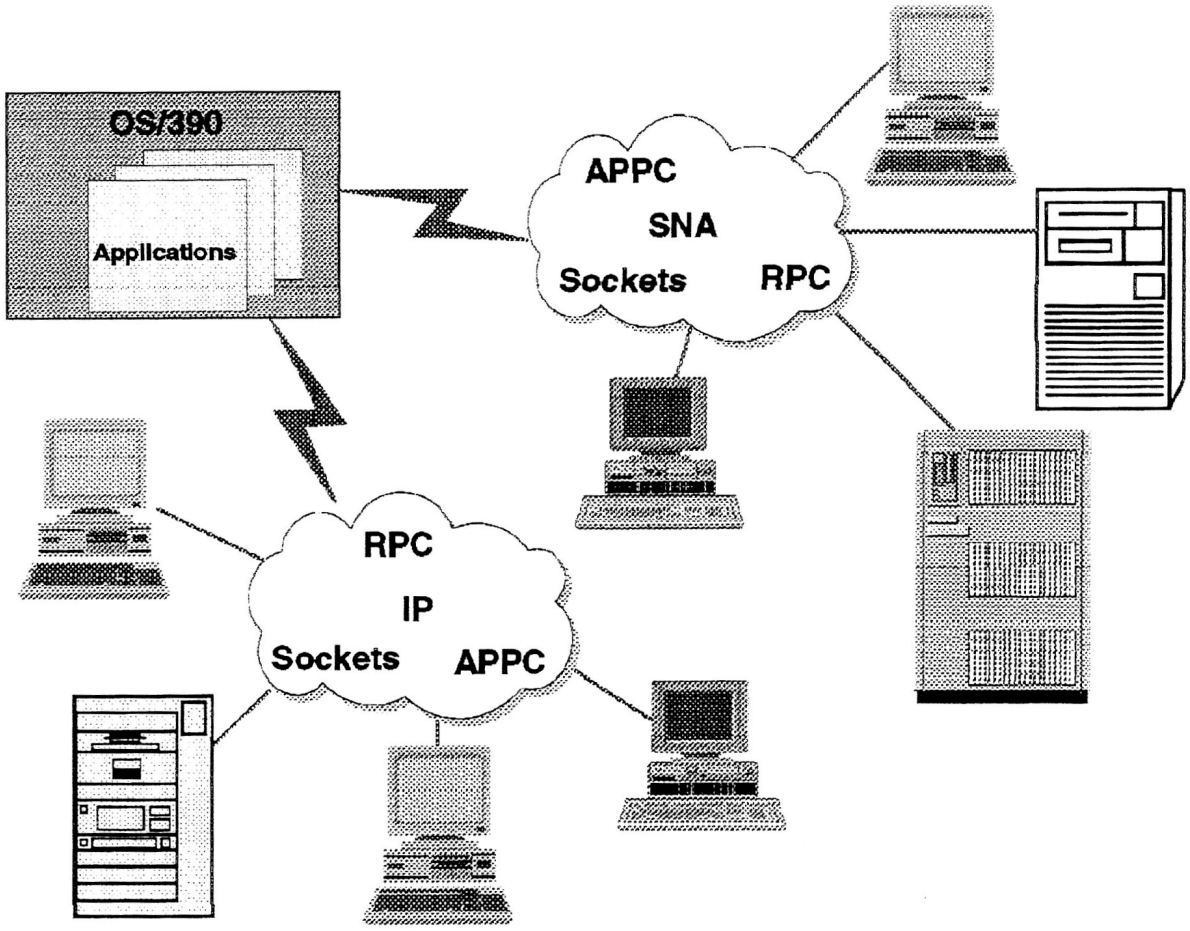
IMS Datenbanksystem

IMS besteht aus zwei Komponenten, dem Database Manager (IMS DM bzw. DB) und dem Transaction Manager (IMS TM).

Das hierarchische Datenmodell des IMS besteht aus einer geordneten Menge von Bäumen, genauer aus Ausprägungen von Bäumen eines bestimmten Typs. Jeder Baum-Typ enthält eine Basis (Root-Segment) und keinen, einen oder mehrere Unterbaum-Typen. Der Unterbaum-Typ seinerseits enthält wiederum eine Basis und ggf. Unterbäume. Die Basis ist jeweils ein logischer Satz (Segment) bestehend aus einem oder mehreren Feldern (Fields).

IMS gestattet die physische und logische Anordnung von Segmenten zu entsprechenden physischen und logischen Datenbanken.

Das obige Beispiel zeigt im oberen Teil einen Baum-Typen und im unteren Teil eine Ausprägung dieses Baum-Typs, einen Kurs, seine Termine und seine dazugehörigen Buchungen.



OS/390 Communication Server

Der OS/390 Communication Server ist ein eigenständiges Subsystem in einem eigenen virtuellen Adressenraum. Er implementiert die TCP/IP und SNA Netzwerk Architektur Stacks.

z/OS “SecureWay” Security Server

- **LDAP Server - Secure Directory Server**

- **Kerberos Network Authentication Service, einschliesslich Kryptographie und Firewall Unterstützung. Hierzu Hardware Unterstützung:**
    - **Zusätzliche Maschinenbefehle für kryptographische Operationen**
    - **Kryptographie Assist-Prozessoren**

- **RACF**

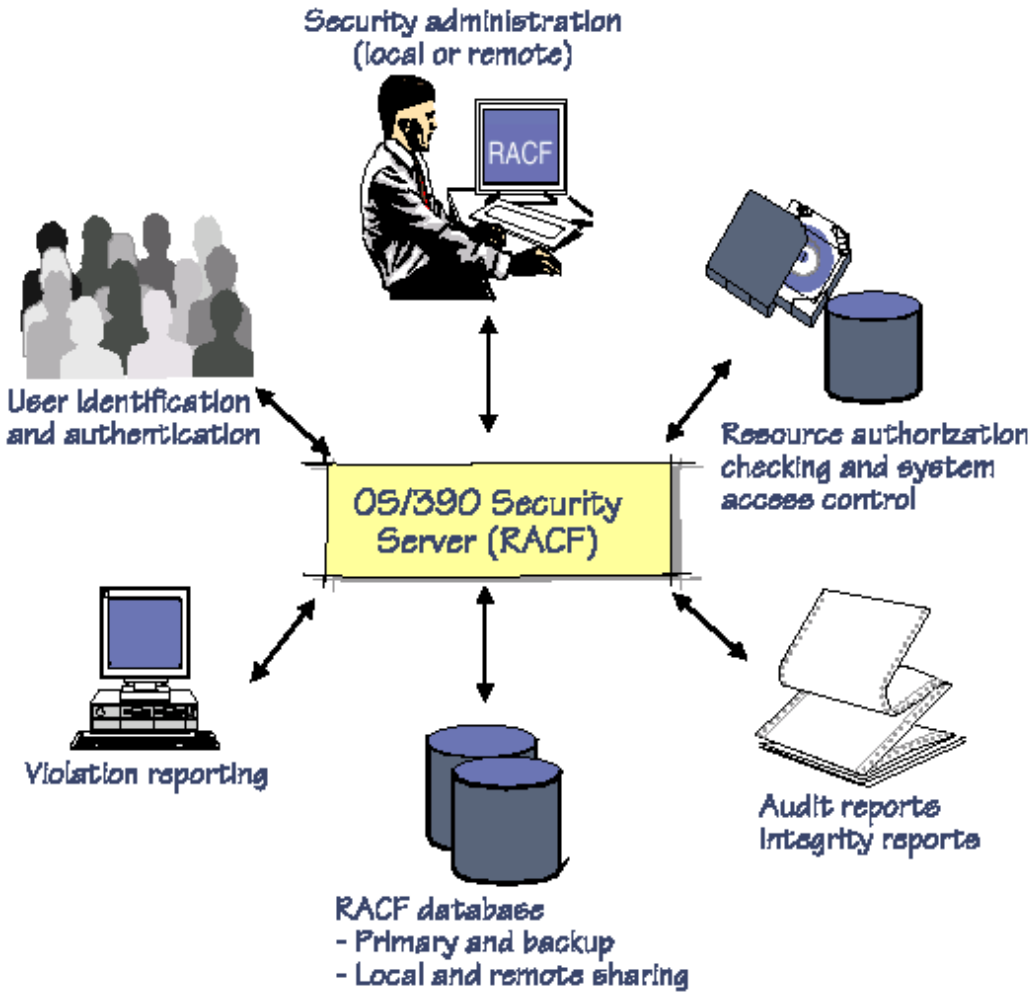
RACF
Resource Access Control Facility

RACF entscheidet,

- **ob der Benutzer für RACF definiert wird**
- **ob der Benutzer ein gültiges Password , PassTicket, Operator Identification Card und einen gültigen Gruppen-Namen verwendet.**
- **ob der Benutzer das System an diesem Tag und zu dieser Tageszeit benutzen darf**
- **ob der Benutzer authorisiert ist, auf das Terminal (Tag und Zeit) zuzugreifen**
- **ob der Benutzer authorisiert ist, auf die Anwendung zuzugreifen**
- **ob der Benutzer authorisiert ist, auf spezifische Daten zuzugreifen**

zSeries bzw. S/390 ist die einzige Rechnerarchitektur auf dem Markt, die über Hardware Speicherschutzschlüssel verfügt.

RACF



RACF bewirkt:

- **Identifizierung und Authentifizierung von Benutzern**
- **Benutzer Authorisierung für Zugriff auf geschützte Resourcen**
- **Logging und Berichte über unauthorisierte Zugriffe**
- **Überwacht die Art, wie auf Resourcen zugegriffen wird**
- **Anwendungen können RACF Macros benutzen**
- **Audit Trail**

Literatur : IBM Form No. GC28-1912-06

RACF

RACF benutzt das Konzept von zu schützenden„Resourcen“.

Resourcen werden in „Klassen“ aufgeteilt. Beispiele für Klassen sind:

- **Benutzer**
- **Dateien**
- **CICS Transaktionen**
- **Datenbank Rechte**
- **Terminals**

Jedem Element einer Klasse wird ein „Profil“ zugeordnet. Das Profil besagt, welche sicherheitsrelevanten Berechtigingen vorhanden sind. Die Profile werden in einer Systemdatenbank, der RACF Profildatenbank, abgespeichert.

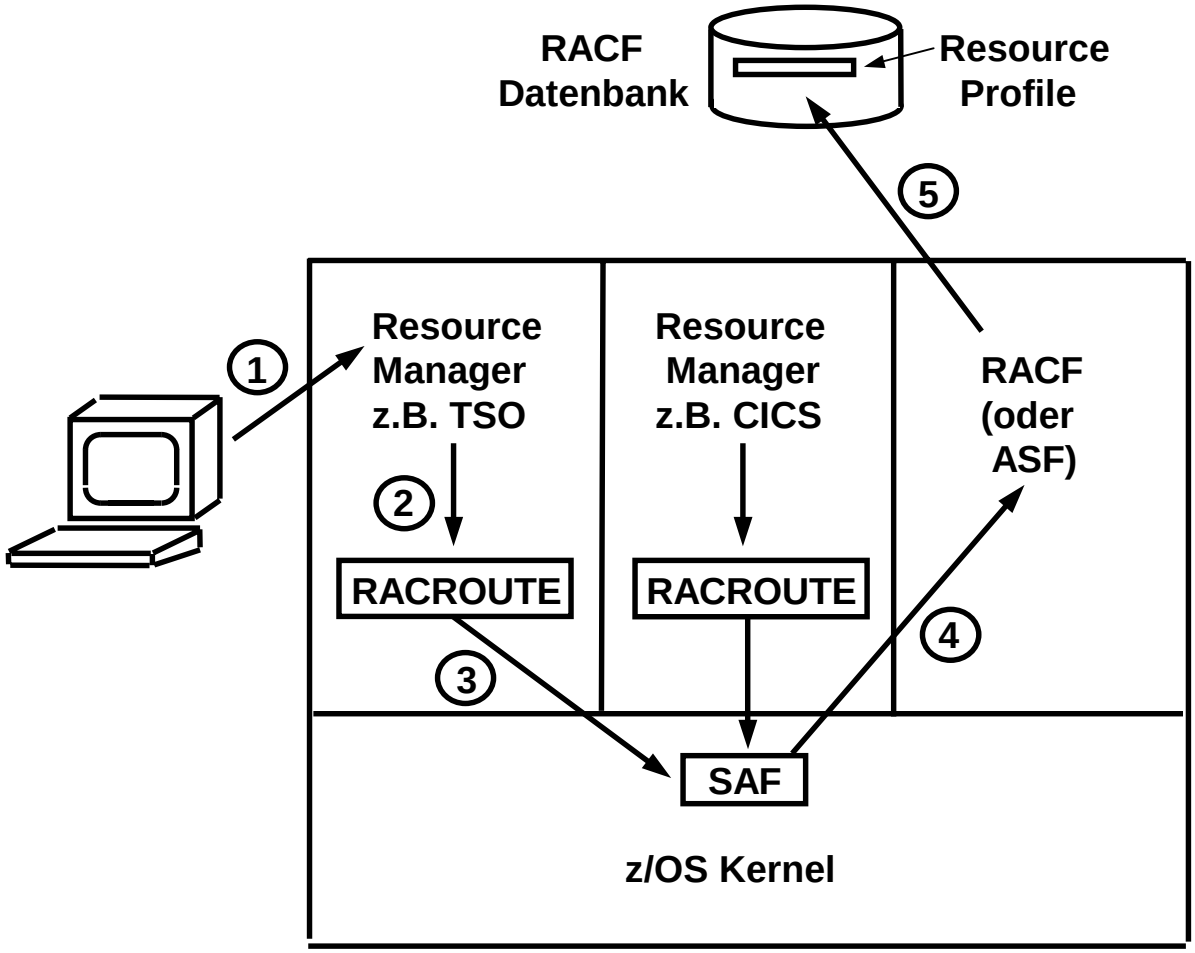
Beispiel: Ein interaktiver Benutzer logged sich ein. Der Login Prozess überprüft, ob die Login Berechtigung besteht. Er überprüft weiterhin, ob das Terminal, von dem der Benutzer sich einwählt, eine Zugangsberechtigung hat. Hierzu ruft der Login Prozess RACF auf, welches die entsprechenden Profile in seiner Datenbank konsultiert.

Anschließend ruft der Benutzer ein Programm auf, welches auf eine Datei zugreift. Die OPEN Routine ruft RACF auf, welches das Profil der Datei bezüglich der Zugriffsrechte befragt.

Benutzer Profile enthalten „Capabilities“. Datei Profile enthalten „Access Control Listen“.

Zugriffsrechte können von spezifischen granularen Bedingungen abhängig gemacht werden; z.B. der Zugriff auf eine Datenbank kann von der Nutzung eines spezifischen Anwendungsprogramms abhängig gemacht werden, oder auf bestimmte Tageszeiten begrenzt sein.

Problem: Wartung der Profile.

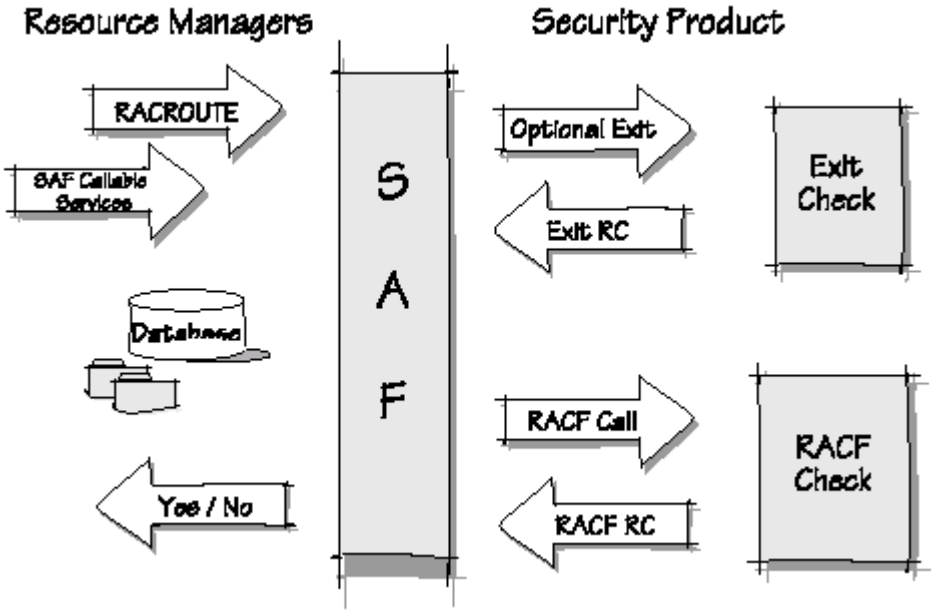


RACF Arbeitsweise

1. Ein Benutzer greift auf eine Resource über einen Resource Manager zu, z.B. TSO
2. Der Resource Manager benutzt einen System Call „RACROUTE“ um auf die Security Access Facility (SAF) des z/OS Kernels zuzugreifen. SAF ist eine zentrale Anlaufstelle für alle sicherheitsrelevanten Aufgaben.
3. SAF ruft ein Zugriffskontrolle Subsystem auf. Dies ist normalerweise RACF. (Eine Alternative ist die Access Control Facility der Fa. Computer Associates, die ähnlich arbeitet).
4. RACF greift auf einen „Profile“ Datensatz in seiner eigenen RACF Datenbank zu und überprüft die Zugangsberechtigungen
5. Das Ergebnis teilt RACF dem anfragenden Resource Manager mit.

RACF (1)
Resource Access Control Facility

OS/390 spezifiziert kritische Events innerhalb des Betriebssystems als sicherheitssensitive Verzweigungen. Die MVS Security Authorisation Facility (SAF) des OS/390 Kernels bewirkt an diesen Stellen den Aufruf einer Security Engine, eines Prozesses, der im Benutzer Status läuft.



In OS/390 ist diese Security Engine häufig RACF; alternative OS/390 Security Engines sind ACF/2 oder TopSecret der Firma Computer Associates.

Zugriffsrechte können von spezifischen granularen Bedingungen abhängig gemacht werden; z.B. der Zugriff auf eine Datenbank kann von der Nutzung eines spezifischen Anwendungsprogramms abhängig gemacht werden, oder auf bestimmte Tageszeiten begrenzt sein.

SAF übergibt der externen Security Engine die pertinente Information. Die Security Engine kann dann auf der Basis von „Profilen“ über die Zugriffsrechte entscheiden.

Problem: Maintenance der Profile.

Supervisor Calls

SVC's (Supervisor Calls) sind das Äquivalent zu den Unix System Calls.

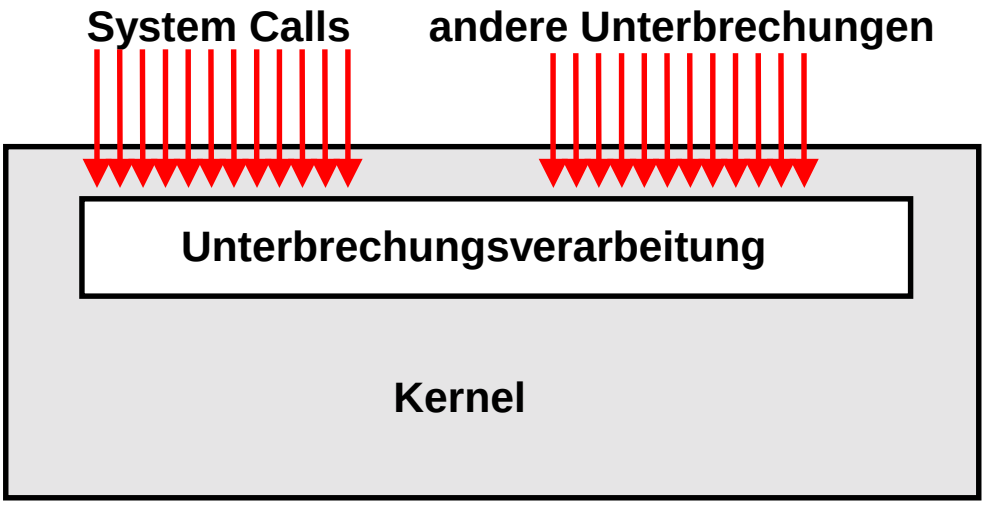
Supervisor Calls im weiteren Sinne sind Library Routinen, die eine Dienstleistung des z/OS Kernels in anspruch nehmen.

Supervisor Calls im engeren Sinne sind Maschinenbefehle, die über einen Übergang vom User Mode (Problem Status) zum Kernel Mode (Supervisor Status) einen Interrupt Handler des Kernels aufrufen. Bei der IA32 (Pentium) Architektur sind dies die INT und CALLGATE Maschinenbefehle.

Ein SVC Maschinenbefehl enthält einen 8 Bit Identifier, welcher die Art des Supervisor Calls identifiziert.

Beispiele sind:

GETMAIN	**SVC 10**	**Anforderung von Virtual Storage**
OPEN	**SVC 19**	**Öffnen eines Data Sets**
EXCP	**SVC 0**	**Lesen oder Schreiben von Daten**
WAIT	**SVC 19**	**Warten auf ein Ereignis, z.B. Abschluß einer Lese Operation**



Die Kernel aller Betriebssysteme haben de facto identische Funktionen, z. B.

- **Unterbrechungsverarbeitung**
- **Prozessmanagement**
- **Scheduling/Dispatching**
- **Ein/Ausgabe Steuerung**
- **Virtuelle Speicherverwaltung**
- **Dateimanagement**

Ein Unix Kernel unterscheidet sich von einem Windows Kernel durch die Syntax und Semantik der unterstützten System Calls, seine Shells sowie durch die unterstützen Dateisysteme.

Linux, Solaris, HP-UX und AIX haben unterschiedliche Kernel, aber (nahezu) identische System Calls..

Die Unix System Services (USS) des z/OS Betriebssystems sind eine Erweiterung des z/OS Kernels um 1100 Unix System Calls, zwei Unix Shells und zwei Unix Datei-systeme. Damit wird aus z/OS ein Unix Betriebssystem.

OS/390 vs. Unix

OS/390

mehrere 1000 E/A Operationen / s

Unix

mehrere 100 E/A Operationen / s

Native Unix Betriebssysteme für S/390

Amdahl UTS (Universal Time Sharing System)
Marktführer, < 300 Installationen

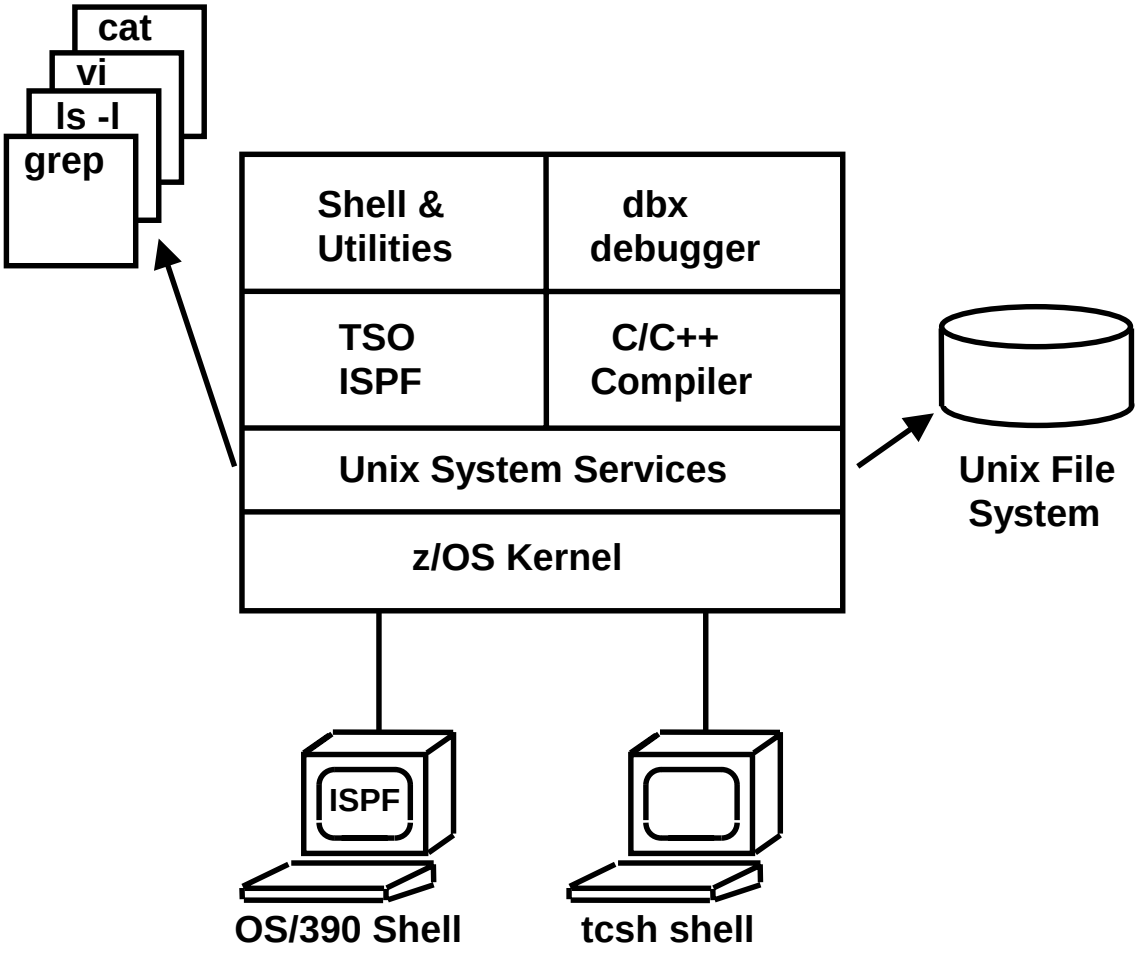
Hitachi HI-OSF/1-M

IBM AIX/ESA

OS/390 Unix System Services

früher als Open Edition MVS bezeichnet

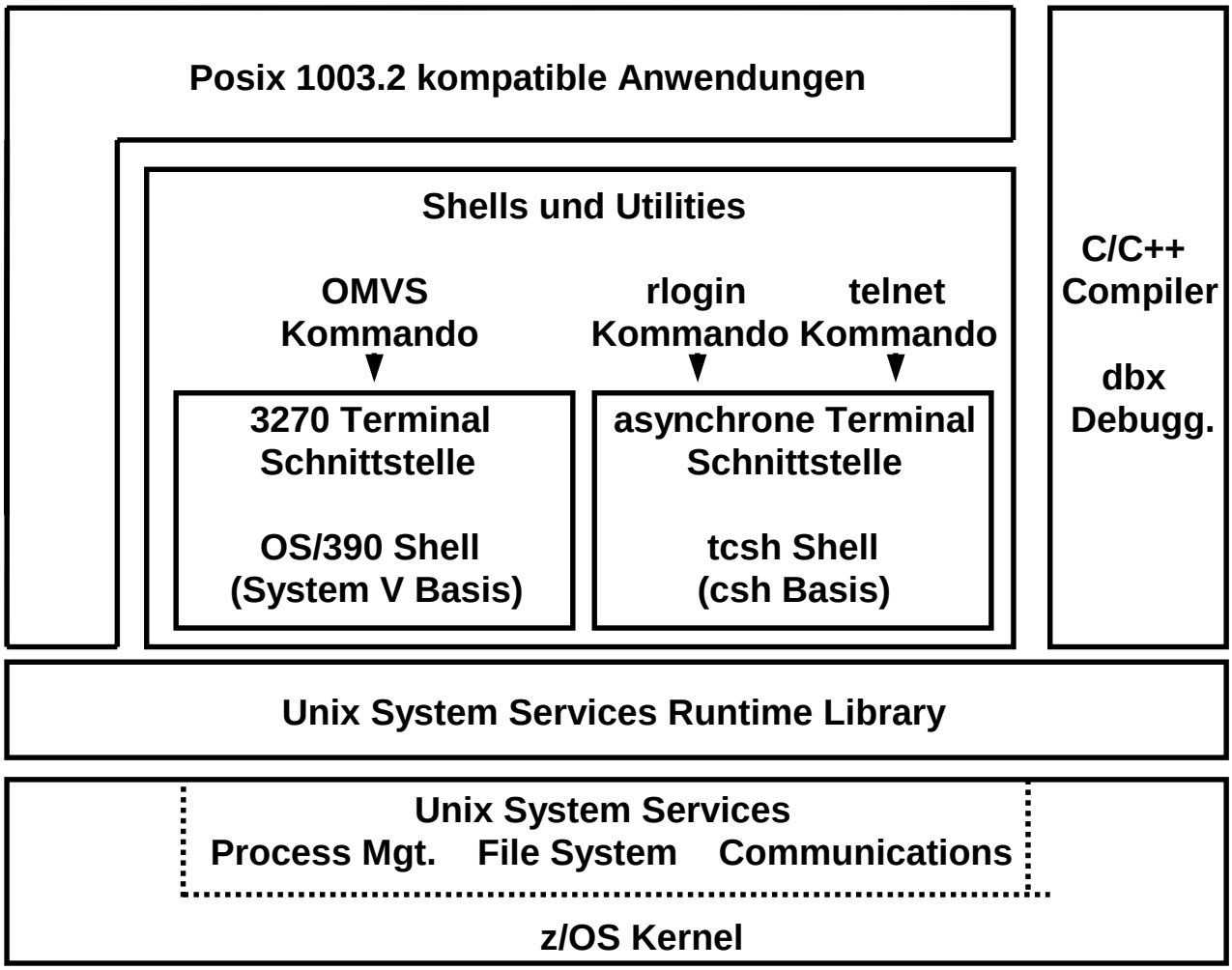
1100 Unix API´s



z/OS Unix System Services verfügt über zwei unterschiedliche Shells.

Die OS/390 Shell gleicht der Unix System V Shell mit einigen zusätzlichen Eigenschaften der Korn Shell. Sie wird meistens von TSO aus über das OMVS Kommando aufgerufen und benutzt den ISPF Editor.

Die tcsh Shell ist kompatibel mit der csh Shell, der Berkley Unix C Shell. Sie wird über rlogin oder telnet aufgerufen und verwendet den vi Editor.



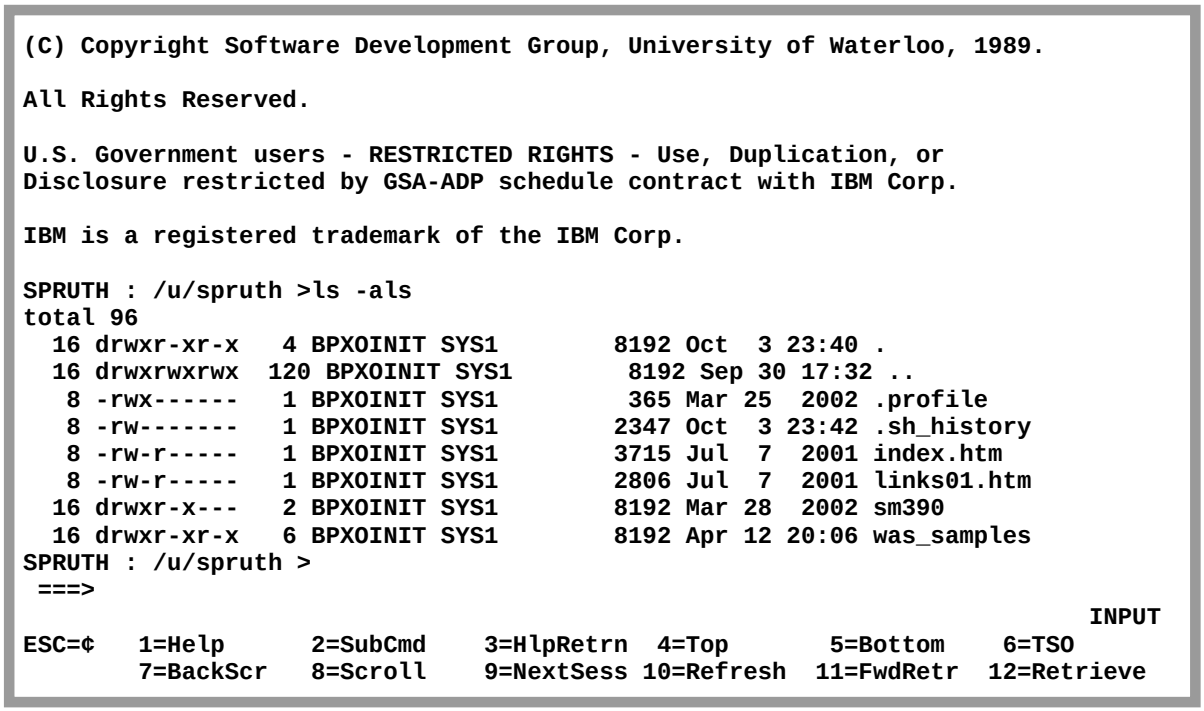
Unix System Services (USS)

z/OS Unix System Services Kernel Funktion läuft in einem eigenen virtuellen Adressenraum, der als Teil des IPL Vorgangs hochgefahren wird. Er wird wie jeder Unix Kernel über eine API aufgerufen, die aus C/C++ Function Calls besteht.

Das Byte-orientierte hierarchische File System arbeitet wie jedes Unix File System. Es wird in z/OS Data Sets abgebildet. Alle Files sind dem Unix System Services Kernel zugeordnet, und alle Ein-/Ausgabe Operationen bewirken Calls in den Kernel.

Die tsch Shell wird normalerweise über rlogin aufgerufen.

Es ist nicht unüblich, auf dem gleichen S/390 Rechner Unix System Services und zLinux parallel zu betreiben.



z/OS Unix System Services Shell

z/OS Unix System Services verfügt über zwei unterschiedliche Shells.

Die OS/390 Shell gleicht der Unix System V Shell mit einigen zusätzlichen Eigenschaften der Korn Shell. Sie wird meistens von TSO aus über das OMVS Kommando aufgerufen und benutzt den ISPF Editor.

Die tcsh Shell ist kompatibel mit der csh Shell, der Berkley Unix C Shell. Sie wird über rlogin oder telnet aufgerufen und verwendet den vi Editor.